Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Franc Oven

Detekcija ovir z zlivanjem senzorske informacije za avtonomna plovila

diplomsko delo

univerzitetni študijski program prve stopnje
računalništvo in informatika

Mentor: doc. dr. Matej Kristan

Ljubljana 2014

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Tematika naloge:

Analizirajte problem detekcije ovir za namen aplikacije na avtonomnem plovilu. Predpostavite, da je avtonomno plovilo opremljeno s sistemom stereo kamer in inercijskim senzorjem. Izpeljite model za zlivanje informacije o oceni horizonta odčitanega z inercijskega senzorja in vizualne informacije pridobljene s kamero. Predlagani postopek naj bo dovolj splošen za aplikacijo tako na monokularni kot stereo kameri. Postopek ovrednotite na zbirki pridobljeni z realnim plovilom.

Izjava o avtorstvu diplomskega dela

Spodaj podpisani Franci Oven, z vpisno številko **63100314**, sem avtor diplomskega dela z naslovom:

Detekcija ovir z zlivanjem senzorske informacije za avtonomna plovila

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom doc. dr. Mateja Kristan,
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela,
- soglašam z javno objavo elektronske oblike diplomskega dela na svetovnem spletu preko univerzitetnega spletnega arhiva.

V Ljubljani, dne 15. septembra 2014 Podpis avtorja:

Zahvaljujem se doc. dr. Mateju Kristan za neutrudljivo pomoč pri izdelavi diplomske naloge, tako pri teoretičnem delu kot pri implementacijskih problemih. Poleg tega še za mnoge sestanke in prediskusirane ideje in obenem tudi za navdušitev za nadaljni študij. Prav tako bi se rad zahvalil tudi moji družini, za podporo in pomoč v času študija.

Kazalo

Povzetek

Abstract

1 Uvod **1**

1.1 Motivacija 1

1.2 Sorodna dela 2

1.3 Cilji in prispevki 3

1.4 Sestava diplomskega dela 4

2 Teoretična podlaga **5**

2.1 Segmentacija slike z Markovim slučajnim poljem 5

2.2 Detekcija objektov 8

2.3 Geometrijski model kamere 9

2.4 3D inferenca 11

3 Zlivanje informacije inercijske merilne naprave in kamere **19**

3.1 Izpeljava preslikave 19

3.2 Ocena horizontalne linije 24

3.3 Upoštevanje horizonta 27

3.4 Zlivanje sterea in segmentacije 32

4 Eksperimentalno vrednotenje **35**

4.1 Anotacija hiperpriorjev 35

4.2 Ocena horizonta 38

4.3 3D inferenca 48

5 Sklep **53**

5.1 Smernice za nadaljnji razvoj 54

Seznam uporabljenih kratic

kratica	angleško	slovensko
HMM	hidden markov models	skrita markova polja
EM	expectation maximization	ekspektacija maksimizacija
PDF	probability density function	verjetnostna porazdelitvena
GMM	gaussian mixture models	mešanica gaussovih porazdelitev
SBM	stereo block matching	stereo bločno ujemanje
P	projection matrix	projekcijska matrika
K	intrinsic matrix	intrinzična matrika
R	rotation matrix	rotacijska matrika
IMU	inertial measurement unit	inercijska merilna naprava
MRF	markov random fields	markova slučajna polja

Povzetek

V diplomski nalogi obravnavamo problem avtonomne navigacije plovil. Predpostavljamo, da je plovilo opremljeno s senzorji kot so GPS, IMU, kompas, stereo sistem dveh kamer ter procesno enoto na kateri se izvajajo algoritmi segmentacije in planiranja poti. Začnemo s predstavitvijo algoritma segmentacije, ki skupaj s pomočjo naprednih metod računalniškega vida iz slik kamer oceni morebitne ovire (boje, plavalci, ladje, obala). Osredotočimo se na del, kjer s pomočjo meritev inercijske merilne naprave ocenimo horizont na slikah kamer in s tem pripomoremo k robustnejši oceni morja. Za izboljšano lokalizacijo ovir pred ladjico, s pomočjo kalibriranega stereo sistema kamer izračunamo globinsko sliko, ki nam pove oddaljenost slikovnih elementov na sliki v tridimenzionalnem prostoru. Slike morajo biti za računanje globine rektificirane, saj s tem poenostavimo iskanje korespondenc na enodimenzionalen problem. V nadaljevanju predstavimo, implementiramo in ovrednotimo naše predloge za izboljšavo ter predstavimo smernice za nadaljni razvoj.

Ključne besede: stereo sistem, 3D inferenca, inercijska merilna naprava, horizont, segmentacija .

Abstract

In our diploma thesis we issue a navigation problem of autonomous surface vehicles. Presume we have a vessel appointed with sensors, such as GPS, IMU, compass, a stereo sistem of two cameras and a standalone processing unit that combines sensor data and performs segmentation and planning in real time. We begin by introducing the algorithm of segmentation, that along with help of advanced computer vision methods extracts useful visual information, therefore avoids obstacles in its path (boats, swimmers, shore, buoys). Then, we focus on a part, where we estimate a horizon line by taking into account data of innertial measurement unit, with whom we improve the estimation of a sea border. To improve localization of obstacles in front of the vessel, we calculate depth image which tells us depth of every pixel in the image. Image pairs are first rectified as we simplify the correspondence search to single dimension. Furthermore, we present, implement and evaluate our methods. We conclude by discussing further work that can be done.

Keywords: stereo system, 3D inference, inertial measurement unit, horizon, segmentation.

Poglavje 1

Uvod

1.1 Motivacija

Avtonomna vozila so že uveljavljena na področjih kot je kontrola mejnih prehodov [13] na mestih, ki so človeku težko dostopna. Uporaba pa ni omejena le na vojaške aplikacije, saj lahko njihovo vlogo razširimo tudi na reševanje industrijskih problemov, kot na primer Amazon Prime Air [34], kjer avtonomno vozilo dostavlja naročene pakete. S tem se zniža cena dostave paketa, naročnik pa lahko paket prejme z minimalnim časovnim zamikom. Prav tako ste najverjetneje že slišali za projekt Google Cars [2], kjer avtomobil brez interakcije voznika uspešno rešuje realne probleme, kot je izogibanje oviram, zaznavanje pešcev in kolesarjev, prepoznavanje znakov in upoštevanje hitrostnih omejitev. V naši diplomski nalogi pa se osredotočimo na primer avtonomnih vodnih vozil kjer imamo sistem, ki v dinamičnem okolju zbira vzorce vode, pri tem pa se z zlivanjem informacije raznih senzorjev (GPS, inercijska merilna naprava, kompas) skupaj s kamero ter svojo procesno enoto, uspešno izogiba oviram (npr. boje, čolni, plavalci) in nemoteno opravlja zadano nalogo. Na Sliki 1.1 je predstavljen sistem take ladjice opremljene s kamerami, inercialnimi senzorji ter svojo procesno enoto.

Slika 1.1: Primer opremljene ladjice.

1.2 Sorodna dela

Detekcija objektov predstavlja pereč problem pri navigaciji z avtonomnimi vozili. Za sledenje poti so avtorji [28] predlagali uporabo vsesmerne (angl., omnidirectional) kamere in z njeno informacijo najdejo pravo pot, vendar ne naslavljajo problema pojavitve dinamičnih objektov. Poleg tega se večina kopenskih avtonomnih vozil pri detekciji objektov zanaša na zaznavo talne površine in globinske senzorje [9, 26], kar pa pri vodnih vozilih lahko privede do napake v meritvah, saj v območju, kjer se nahaja voda, senzor globine nima odziva. Scherer in ostali [29] so predlagali algoritem za detekcijo vode s pomočjo kamere na majhnem helikopterju, ki leti preko reke ter z uporabo vizualne informacije kamere in vgrajenimi senzorji (GPS, inercijska naprava) zgradi tridimenzionalen prostor reke, ki jo je preletel. Možnost integracije senzorjev so pokazali avtorji članka [14], kjer so z uporabo inercialnih senzorjev natančno rekonstruirali trajektorijo vozila, ter avtorji [17], kjer so višino zračnega plovila določili z zlivanjem informacije kamere in senzorjev. Postopek predpostavlja, da se plovilo nahaja v urbanem okolju, kjer se s pomočjo

stavb določijo vertikalne linije, ki sodelujejo pri oceni višine plovila.
Zaradi negotovosti in omejitev senzorjev (predvsem hitrost skeniranja globinskega senzorja), so se avtorji v nadaljevanju za detekcijo ovir in premikajočih se objektov odločili uporabiti kamero. Tako so v [35] uporabili statično kamero in aplicirali algoritem izvzemanja ozadja (angl., background subtraction) skupaj z gibalnimi polji (angl., motion cue), vendar so predpostavke, ki veljajo za algoritem ozadja preveč omejene, da bi postopek uporabili v dinamičnih scenah obalnega pasu. Wang in ostali [36] so za detekcijo večjih objektov na gladini vode uporabili nizko resolucijsko monokularno kamero, kjer so sprva detektirali horizontalno linijo in šele nato iskali potencialne ovire pod horizontom. Tak pristop je uporaben pri nadzorovanju odprtega morja [10], a ker se meja morja določa izključno s horizontom, to privede do problemov v priobalnem pasu, saj rob morja ni več enak horizontalni liniji. Zaradi podobnih problemov so se pojavili bolj splošni segmentacijski pristopi, kot naprimer uporaba optičnega toka, barvne in prostorske informacije za sestavo značilk [18], ki so skupaj uporabljene za segmentacijo objekta iz posnetka. Felzenszwalb in Huttenlocher [11] sta predstavila postopek segmentacije slike na vizualno koherentne regije s pomočjo grafov. Poleg tega so v članku [7] uvedli predpostavko, da sosednji piksli z neko verjetnostjo pripadajo enakemu razredu, kar so naslovili z Markovimi slučajnimi polji.

1.3 Cilji in prispevki

Diplomska naloga vsebuje dva glavna prispevka. Prvi je geometrijski model preslikave meritev naprave IMU v geometrijski sistem kamere. Predlagamo tudi integracijo izpeljanega modela v obstoječi algoritem za semantično razgradnjo slike, ki ga apliciramo na detekcijo ovir. V drugem prispevku predlagamo integracijo predlaganih algoritmov na sistem stereo kamer, kar omogoča lokalizacijo nevarnih ovir v 3D koordinatnem sistemu ladjice. Vse prispevke nato evaluiramo na zbirki, ki smo jo zajeli posebej za ta namen.

1.4 Sestava diplomskega dela

Preostanek diplomskega dela je sledeč. V Poglavju 2 predstavimo teoretično podlago, ki je potrebna za razumevanje problemske domene. Tu predstavimo obstoječi algoritem segmentacije, ki igra ključno vlogo pri avtonomni vožnji ladjice. Nadaljujemo z geometrijskim modelom kamere, ter opisom senzorja IMU. Predstavimo rektificikacijo slikovnega para in opišemo njihovo vlogo pri izračunu disparitete in globinske slike. V Poglavju 3 izpeljemo model preslikave med IMU in kamero z upoštevanjem meritev IMU-ja. Predstavimo način ocenjevanja horizonta in zlivanje teh informacij s kamero. V Poglavju 4 predstavimo našo rešitev za izboljšanje segmentacije z upoštevanjem horizonta ter oceno globine ovir pred ladjico in zaključimo s Poglavjem 5 kjer ovrednotimo naše izboljšave in nakažemo smernice za nadaljni razvoj.

Poglavje 2

Teoretična podlaga

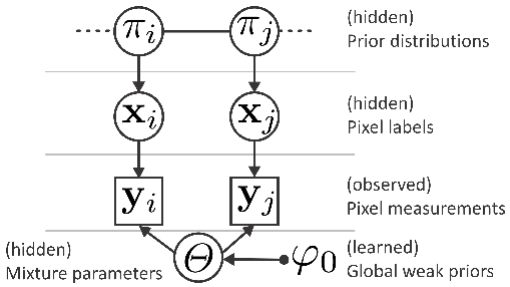
Avtononma vodna vozila (angl., Unmanned Surface Vehicles, USV) zahtevajo poseben pristop pri obravnavi dane problemske domene. Algoritem, predstavljen v nadaljevanju, je fuzija konceptov kot so Markova slučajna polja (angl., Markov Random Fields - MRF), EM (angl., Expectation Maximization) ter GMM (angl., Gaussian Mixture Models). V Podpoglavju 2.1 predstavimo semantični model segmentacije slike, nadaljujemo s Podpoglavjem 2.2, kjer predstavimo algoritem za detekcijo objektov. Nato v Podpoglavju 2.3 predstavimo geometrijski model kamere, od koder nadaljujemo s predstavitvijo rektifikacije slikovnega para, izračunom disparitete in določitvijo globine slikovnih elementov v Podpoglavju 2.4.

2.1 Segmentacija slike z Markovim slučajnim poljem

Obstoječi algoritem segmentacije [25] predstavlja jedro za detekcijo ovir na avtonomni ladjici. Privzamimo, da naša slika vsebuje zbirko meritev $Y = \{y_i\}_{i=1:M}$, kjer y_i v množici $\mathcal{R}^d$ predstavlja d dimenzionalno meritev (vektor značilk), pri i-tem pikslu v sliki z M piksli. Omenjeni vektor značilk je sestavljen iz koordinat posameznega piksla ter njegove barvne reprezentacije. Verjetnost posameznega piksla je predstavljena kot mešanica štirih sestavnih delov – treh Gaussovih in ene uniformne,

$$p(y_i|\Theta) = \sum_{k=1}^{3} \Phi(y_i|\mu, \Sigma)\pi_{ik} + \mathcal{U}(y_i)\pi_{i_4} \,, \tag{2.1}$$

kjer $\Theta = \{\mu_k, \Sigma_k\}_{k=1:3}$ predstavlja srednjo vrednost in kovariance Gaussovih jeder $\Phi(\cdot|\mu, \Sigma)$, $\mathcal{U}$ uniformno distribucijo in $\pi_i = [\pi_{i1}, \ldots, \pi_{i4}]$ apriorno porazdelitev. Gaussove komponente predstavljajo tri glavne semantične regije, ki se pojavljajo v sliki, medtem ko uniformna komponenta predstavlja vsebino, ki ne spada v nobeno izmed omenjenih (angl., outliers) le ta pa nam predstavlja morebitne ovire na vodi, ki se jim je potrebno izogniti. Ker želimo sliko segmentirati v tri približno vertikalno poravnane strukture, se definira začetne parametre nad središči Gaussovih komponent $\varphi = \{\mu_{\mu k}, \Sigma_{\mu k}\}_{k=1:3}$ kar pripelje do sledečega priorja: $p(\Theta|\varphi_0) = \prod_{k=1}^{3} \phi(\mu_k|\mu_{\mu k}, \Sigma_{\mu k})$. Želimo, da so prehodi med regijami kar se da gladki, zato predpostavljamo, da imajo sosednji piksli podobno apriorno porazdeljenost po njihovih razrednih oznakah, kar pomeni, da priorje π_i upoštevamo kot slučajne spremenljivke, ki skupaj tvorijo Markova slučajna polja. Grafični model prikazuje Slika 2.1.



Slika 2.1: Slika semantičnega modela segmentacije. Vir: [25]

Preko vpeljave energijske funkcije za MRF in združitve s semantičnim modelom dobimo funkcijo katere parametri določajo posteriorne verjetnosti pikslov, da pripadajo določeni regiji,

$$F = \sum_{i=1}^{M} [logp(y_i, \Theta|\varphi_o) - \frac{1}{2}(D(s_i||\pi_i \circ \pi_{N_i}) + D(q_i||p_i \circ p_{N_i}))] \,, \tag{2.2}$$

kjer $\circ$ predstavlja Hadamardov produkt [16]. Maksimizacija dane funkcije se nato lahko doseže s postopkom maksimizacije pričakovanja (angl., Expectation Maximization, EM), kjer pri E-koraku maksimiziramo F preko q_i, s_i, medtem ko M-korak

maksimizira preko parametrov Θ in π. Spodnja meja energijske funkcije je definirana z enačbo

$$\hat{F} = \sum_{i=1}^{M} [\frac{1}{2}(q_i + q_{N_i})^T log(p_i p(\Theta|\rho_0)) + \frac{1}{2}(\hat{s}_i + \hat{q}_i)^T log\pi_i] \,. \tag{2.3}$$

Če enačbo (2.3) odvajamo po parametrih Gaussa, ter odvod enačimo z nič, dobimo:

$$\mu_k^{opt} = \beta_k^{-1}[\Lambda_k(\sum_{i=1}^{M} \hat{q}_{ik} y_i^T)\Sigma_k^{-1} - \mu_{\mu_k}^T \Sigma_{\mu_k}^{-1}]^T \,, \tag{2.4}$$

$$\Sigma_k^{opt} = \beta_k^{-1} \sum_{i=1}^{M} \hat{q}_{ik}(y_i - \mu_k)(y_i - \mu_k)^T \,, \tag{2.5}$$

kjer smo rekli da je $\beta_k = \sum_{i=1}^{M} \hat{q}_{ik}$ in $\Lambda_k = (\Sigma_k^{-1} + \Sigma_{\mu_k}^{-1})^{-1}$. Naj bo $\pi_{\cdot k}$ k-ta komponenta priorjev zložena skupaj v velikosti slike. Potem se sosednje priorje lahko izračuna s sledečo konvolucijo $\pi_{N_{\cdot k}} = \pi_{\cdot k} * \lambda$, kjer je λ diskretno jedro s centralnim elementom enakim nič, in vsoto elementov ena. Prav tako uvedimo še $\hat{s}_{\cdot k}, \hat{q}_{\cdot k}$ in $p_{\cdot k}$, ki so enaki velikosti slike in ustrezajo porazdelitvam $\{\hat{s}_i\}_{i=1:M}$, $\{\hat{q}_i\}_{i=1:M}$ in $\{p_i\}_{i=1:M}$ in naj λ_1 predstavlja jedro, kjer je centralni element enak ena. S tem lahko računanje k-te komponente priorjev $\pi_{\cdot k}^{opt}$ za vse piksle v E-koraku zapišemo kot:

$$\begin{aligned} \hat{s}_{\cdot k} &= (\xi_{s\cdot} \circ \pi_{\cdot k} \circ (\pi_{\cdot k} * \lambda)) * \lambda_1 \,, \\ \hat{q}_{\cdot k} &= (\xi_{s\cdot} \circ p_{\cdot k} \circ (p_{\cdot k} * \lambda)) * \lambda_1, \\ \pi_{\cdot k}^{opt} &= (\hat{s}_{\cdot k} + \hat{q}_{\cdot k})/4. \end{aligned} \tag{2.6}$$

Z iteracijo E in M korakov dosežemo konvergenco parametrov k rešitvi[1] (ocenam Gaussovih porazdelitev). Avtorji so pokazali, da se algoritem najboljše izkaže v YCbCr barvnem prostoru [32]. Algoritem svojo hitrost dobi zaradi implementacije z osnovnimi računskimi operacijami kot je množenje in konvolucija, postopek pa prikazuje Algoritem 1.

[1]Podrobno izpeljavo si bralec lahko ogleda v [25]

Algoritem 1 Maksimizacija pričakovanja

Potrebujemo: *Značilke pikslov* $Y=\{y_i\}_{i=1:M}$*, priorji* φ_0*, začetne vrednosti* Θ *in* π*.*

Zagotovi: *Ocenjeni parametri* π^{opt}*,* Θ^{opt} *in glajen posterior* $\{\hat{q}_{\cdot k}\}_{k=1:4}$*.*

1: **Postopek:**

2: *Izračunaj posteriorne porazdelitve pikslov* $p_{\cdot k}$ *z uporabo približkov* π *in* Θ *za vse k po enačbi* (2.1)*.*

3: *Izračunaj nove priorje nad piksli* $\pi_{\cdot k}^{opt}$ *in posteriorje* $\hat{q}_{\cdot k}$ *za vse k po enačbi* (2.6)

4: *Izračunaj nove vrednosti* Θ *po enačbi* (2.4) *in* (2.5)

5: *Ponavljaj postopek 2 do 4 dokler algoritem ne konvergira.*

2.2 Detekcija objektov

Detekcija objektov predstavlja problem, kjer je potrebno iz slike določiti kateri piksli pripadajo morju in kateri ne. Tisti, ki ne pripadajo morju, predstavljajo morebitno oviro, ki jo je potrebno upoštevati pri nadaljnem planiranju poti. Zato se slika najprej pošlje v Algoritem 1 kjer se za vsak piksel izračunajo posteriorne porazdelitve po semantičnih komponentah. Nato se vsi piksli, ki dosežejo maksimum v semantičnem modelu k=1, označijo kot voda. S tem pridobimo sliko B z vrednostmi

$$b_i = \begin{cases} 1 & ; \quad \arg\, max_k\ \hat{\mathrm{q}}_{ik} = 1 \\ 0 & ; \quad \text{otherwise} \end{cases} .$$

Z odstranitvijo vseh regij razen največje dosežemo, da dobimo sliko ovir $\hat{\mathrm{B}}_t$ (angl., obstacle image map). Nato z metodami računalniškega vida izluščimo pomembne podatke kot so rob morja in ovire v vodi. Slednje se določi tako, da v sliki $\hat{\mathrm{B}}_t$ na območju kjer piksli pripadajo vodi, iščemo tiste z vrednostjo nič.

Kot vemo, Algoritem 1 potrebuje začetne vrednosti za Θ in π_t. Na začetku, ko ne poznamo nobenih drugih priorjev, začetno porazdelitev sestavimo z razdelitvijo slike na tri vertikalne regije {0,0.2},{0.2,0.4},{0.6,1}. Iz vsakega dela izračunamo Gaussovo porazdelitev, ki tvori opazovano komponento Θ_{obs}. Priorji vseh pikslov

so nastavljeni na enako verjetnost za vsako regijo, medtem ko so priorji za uniformno komponento nastavljeni na nizko konstantno vrednost. S temi parametri se inicializira EM v Algoritmu 1.

Algoritem 2 Detekcija ovir

Potrebujemo: *Značilke pikslov* $Y=\{y_i\}_{i=1:M}$, *priorji* φ_0, *ocenjen model iz prejšnjega koraka* Θ_{t-1} *in* $\hat{q}_{t-1}$.

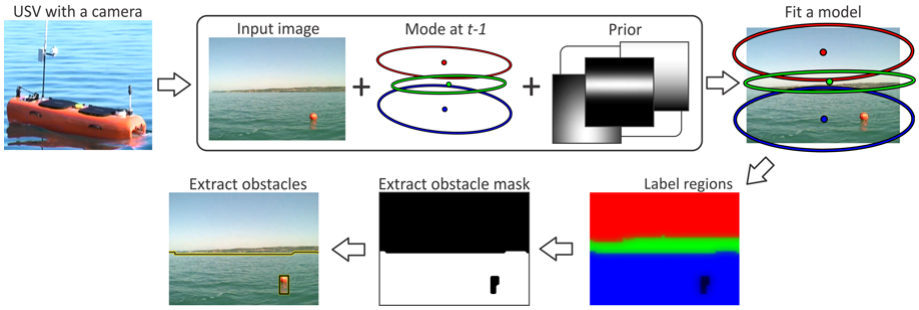
Zagotovi: *Slika ovir* $\hat{B}_t$, *rob morja* e_t, *detektirani objekti* $\{o_i\}_{i=1:N_{obj}}$ *ter parametri modela* Θ_{t-1} *in* $\hat{q}_{t-1}$.

1: **Postopek:**

2: *Inicializiraj parametre* Θ *in* π_t.

3: *Uporabi Algoritem 1 s katerim prilegamo* Θ_t *in* $\hat{q}_t$ *vhodnim podatkom* Y.

4: *Izračunaj novo sliko ovir* $\hat{B}_t$, *rob morja* e_t *in objekte v vodi* $\{o_i\}_{i=1:N_{obj}}$.

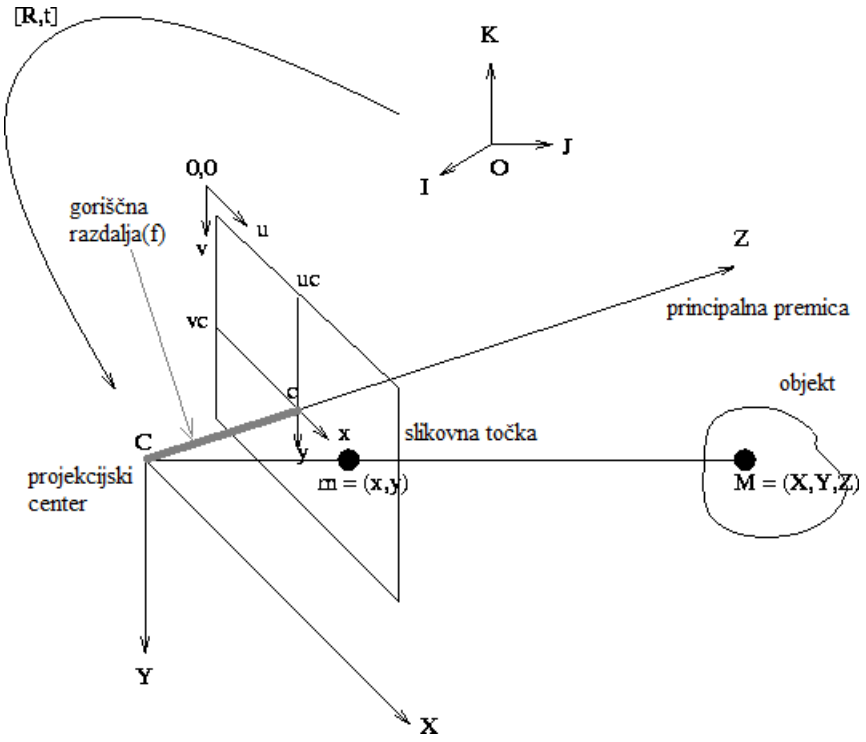


Slika 2.2: Potek algoritma SSM. Vir slike: [25]

2.3 Geometrijski model kamere

Zanima nas, kako se točke v prostoru preslikajo (projecirajo) na slikovno ravnino. Tu je vredno omeniti *centralno projekcijski model* kamere (angl., pinhole camera model with perspective transformation) [12], kjer optično(projekcijsko) središče premaknemo za slikovno ravnino. Zgradbo lepo prikazuje Slika 2.3. Iz nje je razvidno, da kamera uporablja desno sučni koordinatni sistem, kjer koordinata Y kaže navzdol, projekcijski center pa je na sredini slike (oznaka c). Slikovna

ravnina v kameri, kamor se projecirajo točke, je sestavljena iz majhnih senzorjev (pikslov), ki so občutljivi na svetlobo. V idealnih okoliščinah so piksli kamere kvadratni (širina in višina enaki), vendar tovarniške napake povzročajo, da pride do odstopanj. Prav tako tudi pri lečah kamere lahko pride do tako imenovane distorzije, kar povzroči, da se ravne črte v 3D prostoru preslikajo v 2D krivulje. Vpliv le teh se lahko izniči z upoštevanjem distorzijskih parametrov leče, ki se jih pridobi s postopkom kalibracije.



Slika 2.3: Geometrijski model kamere z ladjice, kjer je prikazana preslikava 3D objekta v 2D ravnino. Vir: [1]

Parametri, ki določajo zgradbo kamere in niso odvisni od zunanjega sveta (ali opazovane scene) se imenujejo intrinzični parametri kamere in skupaj sestavljajo kalibracijsko matriko K,

$$K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}, \tag{2.7}$$

kjer (c_x,c_y) predstavljata točko na sliki, kjer principalna premica prebada sliko, f_x in f_y pa predstavljata goriščni razdalji izraženi v enotah pikslov.
Parametri zunanjega sveta, ki se uporabijo za preslikavo iz koordinatnega sistema točke v koordinatni sistem kamere sta rotacijska matrika **R** in translatorni vektor **t**. Skupek teh dveh parametrov predstavljamo v obliki matrike **A**, znano pod imenom matrika zunanjih (ekstrinzičnih) parametrov,

$$A = \Big[R|t\Big] = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} . \tag{2.8}$$

S pomočjo matrike intrinzičnih (2.7) in ekstrinzičnih (2.8) parametrov lahko 3D točko projeciramo v 2D slikovno ravnino:

$$sm' = K \Big[R|t\Big] M' , \tag{2.9}$$

kjer $\mathbf{m}'$ predstavlja 2D točko v homogenem koordinatnem sistemu, $\mathbf{M}'$ 3D točko v homogenih koordinatah, s pa skalirni faktor s tako vrednostjo, da ima tretji element vektorja $\mathbf{m}'$ vrednost 1.

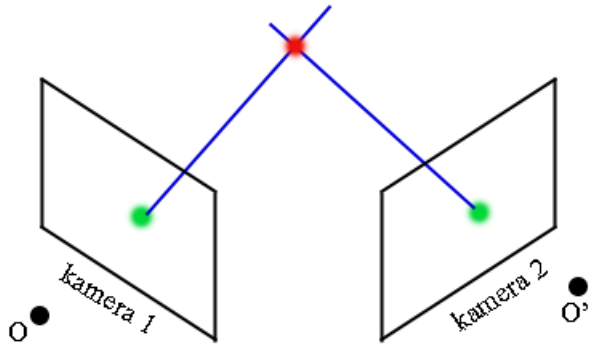
2.4 3D inferenca

Osnovni pristop za določitev globine pri stereo sistemu je triangulacija. Kot je opisano v nadaljevanju, to velja le za zelo preproste stereo sisteme. Globino lahko dobimo če izračunamo presečišče 2D točk kamer v 3D prostoru (Slika 2.4). Za to potrebujemo znano pozicijo kamere v 3D prostoru (imenovano tudi kalibracija, omenjeno v Podpoglavju 2.3) ter korespondence med točkami, to je način preslikave slikovnih točk med levo in desno kamero.

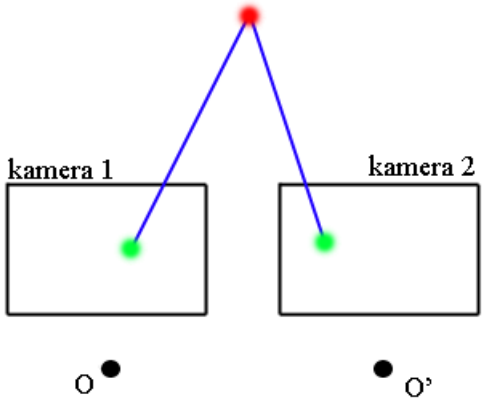
2.4.1 Osnovna geometrija stereo sistema

Predpostavimo, da našo problemsko domeno sestavlja preprost stereo sistem, kjer veljata dve predpostavki:

- Kamere imajo vzporedno poravnane optične osi (za razlago glej Sliko 2.5).
- Poznamo notranje in zunanje parametre kamere (imamo kalibriran sistem).

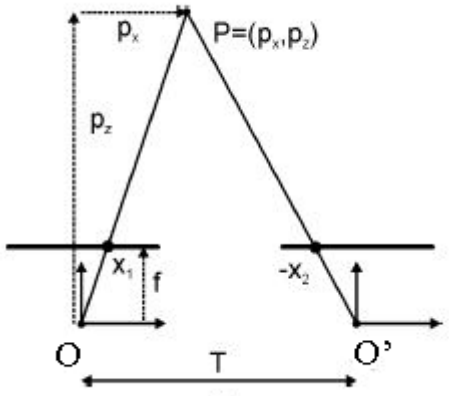


Slika 2.4: Točki leve in desne slike se sekata. O in O' predstavljata centra obeh kamer.



Slika 2.5: Poenostavljen stereo sistem kamer kjer sta optični osi vzporedno poravnani. Izbrano točko v levi kameri lahko najdemo vzdolž iste vrstice v desni kameri.

Ko sta predpostavki resnični, preko trikotnikov (x_1,P,$-x_2$) in (O,P,O') globino točke v kameri izpeljemo s preprosto triangulacijo (Slika 2.6),



Slika 2.6: Vzporedni kameri ki opazujeta točko P.

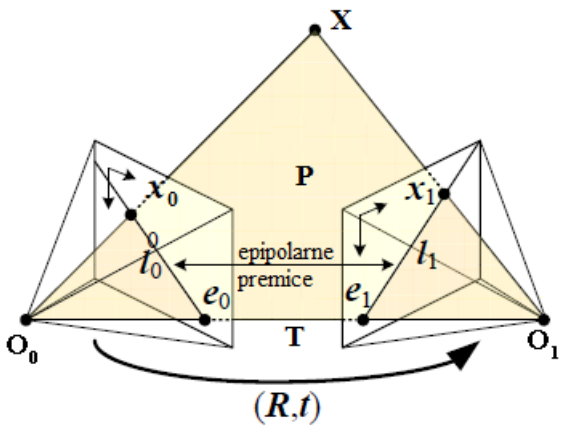
$$\begin{aligned} \frac{T}{pZ} &= \frac{T - x_1 + x_2}{pZ - f}\,, \\ pZ &= f\frac{T}{x_1 - x_2}\,, \\ pZ &= f\frac{T}{D}\,, \end{aligned} \tag{2.10}$$

kjer T predstavlja razdaljo med središčema slik imenovano osnovnico (angl., baseline), f goriščno razdaljo, x_1 in x_2 pa razdaljo od središča slikovne ravnine do točke, kjer P seka slikovno ravnino. Iz enačbe (2.10) opazimo da imamo v imenovalcu $x_1 - x_2$. Ta del predstavlja dispariteto in je podrobno naslovljena v Podpoglavju 2.4.4.

2.4.2 Epipolarna geometrija

Predstavljajmo si, da v 3D prostoru prostoru opazujemo točko X preko dveh kamer, s središčema O_0 in O_1. Točka X se na slikovni ravnini leve kamere preslika v točko x_0, na slikovni ravnini desne kamere pa v x_1. Potem točke O_0,O_1,x_0,x_1 in X ležijo na isti epipolarni ravnini (angl., epipolar plane), označeni s P (Slika 2.7). Posledica koplanarnosti pride prav v primeru ko poznamo slikovno ravnino leve kamere ter preslikavo med O_0 in O_1 in na desni sliki iščemo preslikavo točke X. S tem lahko naše iskanje omejimo le na premico, ki jo določa presečišče epipolarne

ravnine P s slikovno ravnino desne kamere. Tej premici pravimo epipolarna premica in jo poimenujemo z l_0. Različne točke imajo različne epipolarne ravnine, vse pa se sekajo v osnovnici T. Epipolarne premice na vsaki slikovni ravnini se sekajo v točkah e_0 in e_1, ki ju imenujemo epipol.



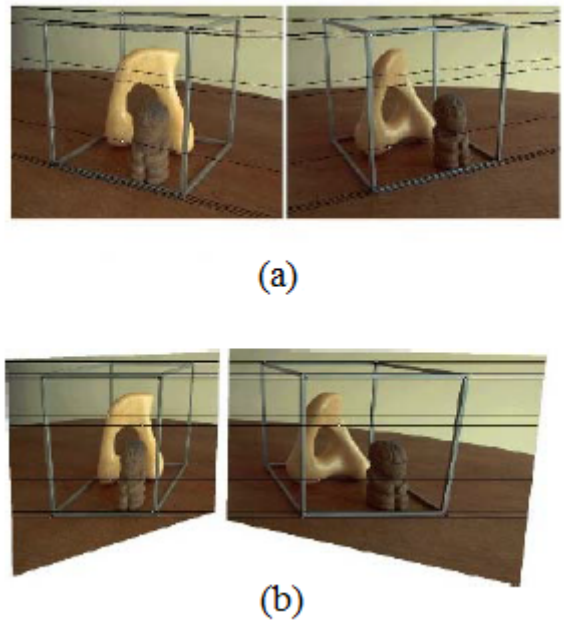
Slika 2.7: Primer epipolarnih premic in epipolarne ravnine pri sistemu dveh kamer. Vir: [33]

2.4.3 Stereo rektifikacija

Postopek računanja globine, kot smo ga opisali v Podpoglavju 2.4.1 v realnosti skoraj ni mogoč, saj kamere skoraj nikoli niso perfektno horizontalno poravnane in tudi če so, sčasoma pride do odstopanj in napake pri izračunu globine postanejo prevelike. S pomočjo epipolarne geometrije je izračun dokaj preprost, vendar so epipolarne premice pogosto nagnjene pod določenim kotom, kar nam še vedno predstavlja veliko računsko zahtevnost pri iskanju podobnosti vzdolž epipolarnih linij.

Če poznamo homografiji za preslikavo leve in desne slikovne ravnine kamer v neko skupno ravnino, ki je vzporedna osnovnici (Slika 2.8), potem točko x_1 (Podpoglavje 2.4.2) v desni sliki iščemo zgolj horizontalno, na enaki višini kot je x_0. To pomeni da ko pogledamo neko točko v levi slikovni ravnini, lahko projekcijo te točke v desni slikovni ravnini najdemo na enaki višini (epipolarne premice so poravnane). Temu postopku pravimo rektifikacija in ga v tem diplomskem delu ne

bomo obravnavali podrobno, bralec pa si lahko več prebere v [22]. Po rektifikaciji epipolarne linije sovpadajo z vrsticami slike. S tem se približamo horizontalni poravnanosti kamer v Podpoglavju 2.4.1.



Slika 2.8: Slika (a) prikazuje par stereo slik skupaj z epipolarnimi linijami, ter slika (b), ki prikazuje rektificiran par slik. Iz slike (b) je razvidno, da epipolarne premice obeh slik sovpadajo. Vir: [33]

2.4.4 Dispariteta

V primeru rektificiranega para slik, se točka x_0 iz leve slikovne ravnine v desni nahaja na enaki višini a ne nujno na enaki koordinati širine. Tej razliki pravimo dispariteta. Najlažje si jo predstavljamo, da zapremo eno oko, in si z drugim izberemo nek predmet blizu nas. Nato istočasno prvo oko odpremo in drugega zapremo. Razliki v lokaciji kjer zaznamo predmet, pravimo dispariteta. Postopek izračuna disparitetne mape delimo v dve skupini:

- Metoda za gosto ocenjevanje (angl., dense),

- Metoda za redko ocenjevanje (angl., sparse).

Razlogov uporabe za oba je veliko, mi bomo našteli le nekaj razlik. Pri metodah za gosto ocenjevanje disparitete lahko rekonstruiramo skoraj celotno prostorsko strukturo opazovane scene. Slabosti so pri intenzitetah slikovnih točk saj niso primerne za ujemanja in pri ocenjevanju disparitete v regijah slike, kjer ni teksture saj se le ta ne da izračunati.
Redko ocenjeno dispariteto uporabljamo v primerih, kadar ne rabimo rekonstruirati celotne opazovane scene. Tu se pogosto uporabljajo detektorji značilk (angl., keypoint descriptors) kot so SIFT [23], SURF [6], BRISK [21] ali FREAK [4] saj je pomembno da se izberejo robustni deskriptorji, ki izberejo točke ki niso občutljive na spremembo intenzitete. Prednost uporabe značilk je višja hitrost izvajanja zaradi manjše množice točk pri iskanju disparitete.

2.4.5 Ocenjevanje disparitete in globine

V naši problemski domeni ocenjujemo gosto dispariteto (Slika 2.9) in nato preko inverza izračunamo globino posameznih pikslov (enačba 2.10). Postopek za izračun disparitete je sledeč. Predpostavljamo, da sta goriščni razdalji kamer f in njuna osnovnica T znani. Imamo stereo par slik, za katere definirajmo točko x_0 v levi sliki in za katero želimo najti korespondenčno točko x_1 v desni sliki. Točko x_1 v desni sliki iščemo vzdolž epipolarne premice, ki je določena z epipolarno ravnino (Slika 2.7). Ponavadi jo iščemo preko obeh dimenzij (širine in višine), iskanje pa omejimo le na okolico, kjer se nahaja x_0. Točka z največjo podobnostjo je izbrana kot korespondenčna točka, in zanjo se izračuna dispariteta,

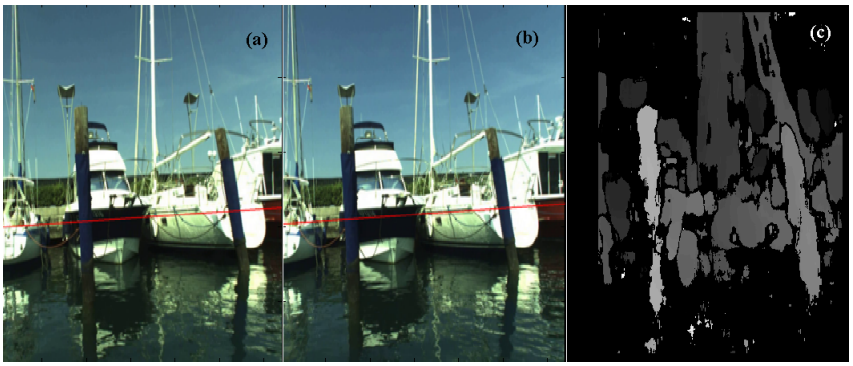
$$d = x_0 - x_1 \, . \tag{2.11}$$

To ponovimo za vse slikovne elemente in s tem zgradimo disparitetno mapo. Nato s pomočjo enačbe (2.10) izračunamo globino vseh slikovnih točk.
V praksi je iskanje korespondenčnih točk v dveh dimenzijah računsko zahtevna operacija, zato se stereo par pred izračunom disparitete rektificira. S tem dosežemo, da epipolarne premice leve in desne slike sovpadajo (višini premic v levi in desni sliki sta enaki) kar pomeni, da se iskanje korespondenčnih točk poenostavi na iskanje v enodimenzionalnem prostoru. Tako lahko korespondenčno točko x_1 v desni

sliki iščemo zgolj po vrstici, v kateri se nahaja točka x_0 v levi sliki. Postopek je povzet v Algoritmu 3.

Algoritem 3 Izračun disparitete/globine

1: **Postopek:**
2: *Za vsak piksel x_i v levi sliki:*
3: *Najdi pripadajočo epipolarno premico v desni sliki.*
4: *Med vsemi točkami na epipolarni premici izberi tisto najbolj podobno x_i (z uporabo SSD ali korelacije). V primeru, da imamo stereo par rektificiran, so epipolarne premice horizontalno poravnane, kar poenostavi iskanje korespondenc na enodimenzionalen problem.*
5: *Izračunaj globino s pomočjo triangulacije* (2.10).



Slika 2.9: Sliki (a) in (b) tvorita stereo par, katerega dispariteta predstavlja slika (c).

Slika 2.10: Rektificirani sliki z epipolarnimi premicami zelene barve. Če pogledamo levo sliko in si na zeleni črti izberemo točko, jo lahko na desni sliki najdemo tako, da sledimo zeleni liniji.

Poglavje 3

Zlivanje informacije inercijske merilne naprave in kamere

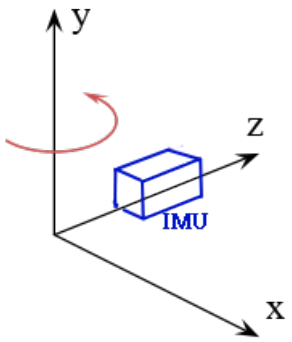
Za upoštevanje inercijskih meritev naprave moramo poznati naslednje: parametre

- Intrinzične parametre kamere ter njene distorzijske koeficiente.
- Translacijo in rotacijo med kamero in IMU-jem.
- Kako se meritve IMU-ja $(roll, pitch, yaw)$ upoštevajo pri preslikavi točk iz svetovnega koordinatnega sistema v koordinatni sistem kamere.

V Podpoglavju 3.1 izpeljemo rotacijsko matriko za senzor IMU glede na koordinatni sistem kamere, nato v Podpoglavju 3.2 predstavimo način za ocenjevanje horizonta v sliki, v Podpoglavju 3.3 pa predstavimo vlogo horizonta pri segmentaciji slike.

3.1 Izpeljava preslikave

Za izpeljavo rotacijske matrike moramo sprva definirati posamezne rotacije, ki skupaj določajo rotacijo v 3D prostoru. Senzor IMU je v ladjico vpet vzdolž Z osi, kot to prikazuje Slika 3.1. Vsako rotacijo v 3D prostoru je moč opisati z zmnožkom rotacij okoli vsake osi (X,Y,Z). Iz preproste geometrije vemo, da lahko 2D rotacijo opišemo kot vsoto sinusnih oziroma kosinusnih produktov. V našem primeru so rotacije okoli osi prikazane na Slikah 3.2, 3.3 in 3.4.



Slika 3.1: Koordinatni sistem IMU naprave sledi pravilu desne roke kjer rotacije okoli posamezne osi potekajo v nasprotni smeri urinega kazalca. Rotaciji okoli osi Z pravimo *roll*, okoli X *pitch*, in okoli osi Y *yaw*.

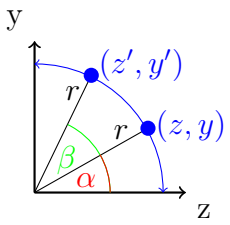
3.1.1 Rotacija po X (pitch)

Denimo, da imamo točko r in jo želimo zavrteti v novo točko r tako, da jo zavrtimo okoli osi X za kot β, kot to prikazuje Slika 3.2. V primeru da točka že leži pod nekim kotom α glede na koordinatno izhodišče, sta enačbi za izračun z in y sledeči:

$$z = r\cos(\alpha)\,, \qquad y = r\sin(\alpha) \tag{3.1}$$

in za izračun z' in y' pri danem kotu β:

$$z' = r\cos(\alpha+\beta)\,, \qquad y' = r\sin(\alpha+\beta)\,. \tag{3.2}$$



Slika 3.2: Prikazuje rotacijo okoli osi X(pitch).

S pomočjo sinusnih in kosinusnih izrekov za dvojne kote lahko z' zapišemo kot:

$$\begin{aligned} z' &= r\cos(\alpha+\beta) \\ &= r(\cos\alpha\cos\beta - \sin\alpha\sin\beta) \\ &= r\cos\alpha\cos\beta - r\sin\alpha\sin\beta \\ &= z\cos\beta - y\sin\beta \end{aligned} \tag{3.3}$$

in y' kot:

$$\begin{aligned} y' &= r\sin(\alpha+\beta) \\ &= r(\sin\alpha\cos\beta + \cos\alpha\sin\beta) \\ &= r\sin\alpha\cos\beta + r\cos\alpha\sin\beta \\ &= y\cos\beta + z\sin\beta\,. \end{aligned} \tag{3.4}$$

Enačbi za z' in y' združimo skupaj v matriko za 3D rotacijo (ker vrtimo okoli osi X, je na njenem mestu enka):

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\beta & \sin\beta \\ 0 & -\sin\beta & \cos\beta \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}\,. \tag{3.5}$$

Rotacijska matrika $R_{pitch}(3\times 3)$ iz enačbe 3.5 predstavlja zasuk okoli osi X.

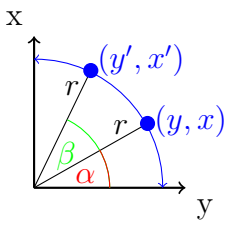
3.1.2 Rotacija po Z (roll)

Tudi tukaj želimo točko r zavrteti v novo točko tako, da jo zavrtimo okoli osi Z za kot β, kot prikazuje Slika 3.3. V primeru, da točka r leži pod danim kotom α, sta enačbi za izračun y in x sledeči:

$$y = r\cos(\alpha)\,, \qquad x = r\sin(\alpha) \tag{3.6}$$

in za izračun y' in x' pri danem kotu β:

$$y' = r\cos(\alpha+\beta)\,, \qquad x' = r\sin(\alpha+\beta)\,. \tag{3.7}$$



Slika 3.3: Prikazuje rotacijo okoli osi Z imenovano tudi *roll*.

Tudi tu s pomočjo sinusnih in kosinusnih izrekov za dvojne kote y' zapišemo kot:

$$\begin{aligned} y' &= r\cos(\alpha+\beta) \\ &= r(\cos\alpha\cos\beta - \sin\alpha\sin\beta) \\ &= r\cos\alpha\cos\beta - r\sin\alpha\sin\beta \\ &= y\cos\beta - x\sin\beta \end{aligned} \tag{3.8}$$

in x' kot:

$$\begin{aligned} x' &= r\sin(\alpha+\beta) \\ &= r(\sin\alpha\cos\beta + \cos\alpha\sin\beta) \\ &= r\sin\alpha\cos\beta + r\cos\alpha\sin\beta \\ &= x\cos\beta + y\sin\beta\,, \end{aligned} \tag{3.9}$$

katere enako kot v prejšnjem Podpoglavju združimo v 3×3 matriko. Ker se tokrat vrtimo okoli osi Z, je na njenem mestu enka.

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} \cos\beta & \sin\beta & 0 \\ -\sin\beta & \cos\beta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{3.10}$$

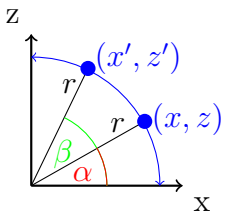
3.1.3 Rotacija po Y (yaw)

V primeru, da želimo točko r, ki leži pod kotom α, zasukati okoli osi Y za kot β (glej Sliko 3.4), sta enačbi za izračun x in z sledeči:

$$x = r\cos(\alpha)\,, \qquad\qquad z = r\sin(\alpha) \tag{3.11}$$

in za izračun x' in z' pri danem kotu β:

$$x' = r\cos(\alpha+\beta)\,, \qquad z' = r\sin(\alpha+\beta)\,. \tag{3.12}$$



Slika 3.4: Rotacija okoli osi Z imenovana tudi yaw.

S pomočjo geometrijskih izrekov za x' sledi:

$$\begin{aligned} x' &= r\cos(\alpha+\beta) \\ &= r(\cos\alpha\cos\beta - \sin\alpha\sin\beta) \\ &= r\cos\alpha\cos\beta - r\sin\alpha\sin\beta \\ &= x\cos\beta - z\sin\beta \end{aligned} \tag{3.13}$$

in z' :

$$\begin{aligned} z' &= r\sin(\alpha+\beta) \\ &= r(\sin\alpha\cos\beta + \cos\alpha\sin\beta) \\ &= r\sin\alpha\cos\beta + r\cos\alpha\sin\beta \\ &= z\cos\beta + x\sin\beta\,. \end{aligned} \tag{3.14}$$

Enačbi zapišemo v 3×3 matriki. Vrtenje okoli Y pomeni, da je na njenem mestu enka.

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} \cos\beta & 0 & -\sin\beta \\ 0 & 1 & 0 \\ \sin\beta & 0 & \cos\beta \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{3.15}$$

Rotacijska matrika $R_{yaw}(3\times 3)$, ki predstavlja zasuk okoli osi Y.

3.1.4 Rotacijska matrika

V Podpoglavjih 3.1.2, 3.1.1, 3.1.3 smo izpeljali rotacijske matrike okoli posameznih osi. V našem primeru sta kameri na ladjici vpeti tako, da gledata vzdolž osi Z, zato za oceno horizonta potrebujemo le zasuk okoli osi $X(roll)$ in zasuk okoli $Z(pitch)$. *Roll* nam nakazuje naklon premice, *pitch* pa opisuje prosti parameter. Zato tu potrebujemo rotacijske matrike R_{roll} (3.10) in R_{pitch} (3.5). Rotacijska matrika, ki igra ključno vlogo je torej

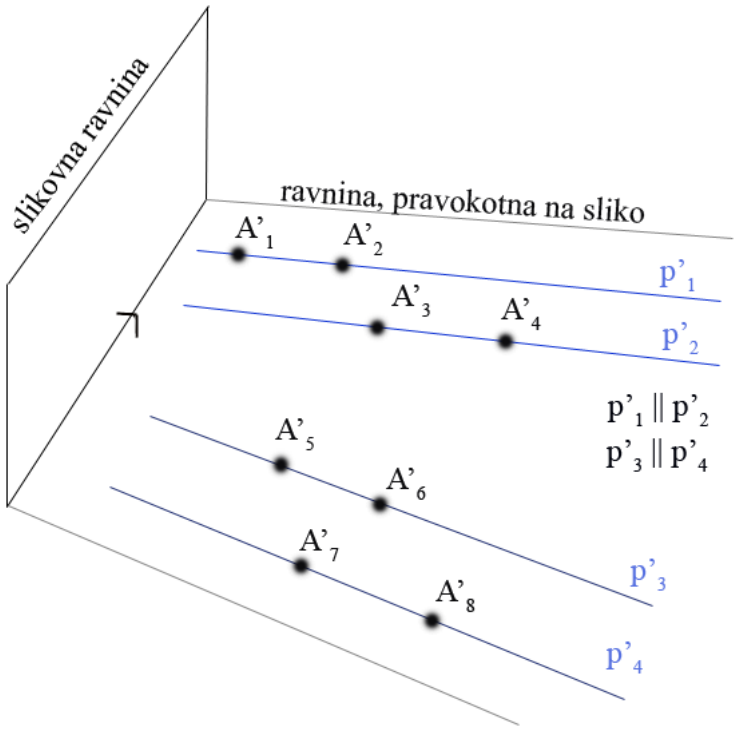
$$R_{IMU} = R_{roll} \cdot R_{pitch} = \begin{bmatrix} \cos\alpha & \sin\alpha\cos\beta & \sin\alpha\sin\beta \\ -\sin\alpha & \cos\alpha\cos\beta & \cos\alpha\sin\beta \\ 0 & -\sin\beta & \cos\beta \end{bmatrix}. \tag{3.16}$$

3.2 Ocena horizontalne linije

V nadaljevanju predstavimo preslikavo točk iz 3D v 2D prostor, kjer pari točk tvorijo premice, te pa se sekajo v horizontu, ki nam predstavlja množico navideznih točk v neskončnosti. Za začetek si poglejmo Sliko 3.5, ki prikazuje slikovno ravnino in pred njo nabor točk. Vsaka točka je predstavljena s koordinatami $(\mathbf{x'_i}, \mathbf{y'_i})$. Kot opazimo, so vse točke od $\mathbf{A1'}$ do $\mathbf{A8'}$ vpete v isti ravnini. Izberimo pare točk $(\mathbf{A1'}, \mathbf{A2'})$, $(\mathbf{A3'}, \mathbf{A4'})$, $(\mathbf{A5'}, \mathbf{A6'})$, $(\mathbf{A7'}, \mathbf{A8'})$ in čez potegnimo premice, ki so označene s $\mathbf{p'_{1..4}}$. Razvidno je, da so višine pri vseh točkah enake saj je pogoj, da so točke v skupni ravnini. Globina točk (koordinata Z) je nenegativna, pri koordinati X pa moramo paziti, da je razmerje med prvo in drugo točko (torej $A1' - A2'$ in $A3' - A4'$) enako. Premice p'_1 in p'_2 ter p'_3 in p'_4 so tedaj v 3D prostoru med seboj vzporedne.

Vse točke nato s pomočjo matrike intrinzičnih in ekstrinzičnih parametrov preslikamo v 2D prostor (Slika 3.6), katere so sedaj opisane s koordinatami $(\mathbf{x_i}, \mathbf{y_i})$ in paroma označujejo premice $\mathbf{p_{1..4}}$. Premice $p_{1..4}$ (preslikane na slikovno ravnino), se sekajo v bežiščih $\mathbf{h_1}$ in $\mathbf{h_2}$ (angl., vanishing point). Iz točk $\mathbf{h_1}$ in $\mathbf{h_2}$ nato izračunamo enačbo premice, ki določa horizont.

Analitično izračunajmo premico, ki je naš približek horizonta. Vsaka naša točka $\mathbf{A'_{1..8}}$ je predstavljena s koordinatami $(\mathbf{x'_i}, \mathbf{y'_i})$. Vse točke s pomočjo enačbe (2.9) preslikamo v 2D prostor in so sedaj opisane s koordinatami $(\mathbf{x_{1..8}}, \mathbf{y_{1..8}})$, rotacijska



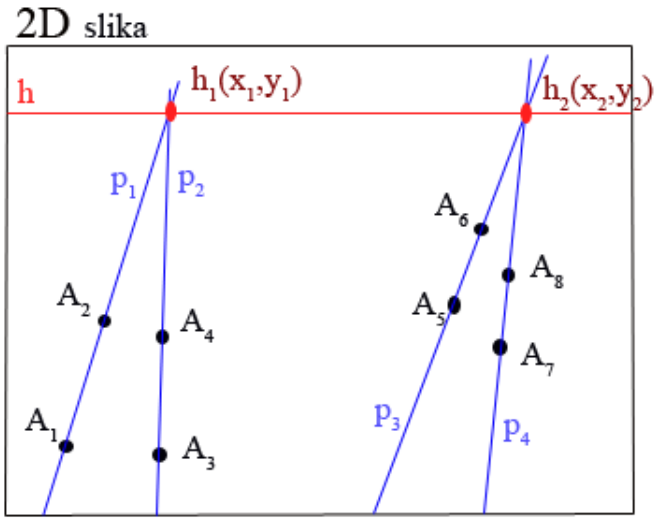
Slika 3.5: 3D točke, katerih pari tvorijo premice. Pari (p'_1,p'_2) ter (p'_3,p'_4) so med vseboj vzporedni

matrika pa je predstavljena z $\mathbf{R_{IMU}}$ iz Podpoglavja 3.1.4 (enačba 3.16). Translacijski vektor t nastavimo na nič. Nato za vsak par preslikanih točk (A1,A2),(A3,A4),(A5,A6),(A7,A8) zapišemo enačbe premic, ki jih določajo. Za izračun premice p_1 imamo dve točki (A1 in A2), ki ju lahko zapišemo v obliki premice in določata parametra k_1 in n_1,

$$y_1 = k_1 x_1 + n_1 \,, \qquad y_2 = k_1 x_2 + n_1 \,, \qquad p_1 = k_1 x + n_1 \,. \tag{3.17}$$

Postopek ponovimo za vse štiri premice $p_{1..4}$ in z izpostavitvijo parametrov dobimo,

$$\begin{aligned}
p_1: &\qquad k_1 = \frac{y_1 - y_2}{x_1 - x_2} &\qquad n_1 &= y_1 - k x_1 \,, \\
p_2: &\qquad k_2 = \frac{y_3 - y_4}{x_3 - x_4} &\qquad n_2 &= y_3 - k x_3 \,, \\
p_3: &\qquad k_3 = \frac{y_5 - y_6}{x_5 - x_6} &\qquad n_3 &= y_5 - k x_5 \,, \\
p_4: &\qquad k_4 = \frac{y_7 - y_8}{x_7 - x_8} &\qquad n_4 &= y_7 - k x_7 \,.
\end{aligned} \tag{3.18}$$



Slika 3.6: 3D premice, ki so med seboj vzporedne, se v 2D prav tako preslikajo v premice, vendar imajo presečišče v horizontu. Črka h nam označuje horizont, h_1 in h_2 pa presečišči premic (p_1, p_2) ter (p_3, p_4).

Premici p_1 in p_2 ter p_3 in p_4 se sekajo v točki $\mathbf{h_1}$ in $\mathbf{h_2}$ s koordinatami (x_{hi}, y_{hi}):

$$\begin{aligned} h_1: \quad & x_{h1} = \frac{n_2 - n_1}{k_1 - k_2} \qquad y_{h1} = k_1 x + n_1 \,, \\ h_2: \quad & x_{h2} = \frac{n_4 - n_3}{k_3 - k_4} \qquad y_{h2} = k_3 x + n_3 \,. \end{aligned} \tag{3.19}$$

Ker točki $h_1(x_{h1}, y_{h1})$ in $h_2(x_{h2}, y_{h2})$ predstavljata sliko točk v navidezni neskončnosti, v sliki kamere ležita na horizontu. Torej lahko enačbo horizonta zapišemo kot

$$\begin{aligned} & y_{h1} = k_h x_{h1} + n_h \qquad y_{h2} = k_h x_{h2} + n_h \,, \\ & k_h = \frac{y_{h1} - y_{h2}}{x_{h1} - x_{h2}} \qquad n_h = y_{h1} - k x_{h1} \,. \end{aligned} \tag{3.20}$$

Izračunana parametra k_h in n_h nam določata premico h, ki je ocena horizonta

$$h = k_h x + n_h \,. \tag{3.21}$$

3.3 Upoštevanje horizonta

Premico horizonta, ki smo jo določili s pomočjo IMU meritev (3.21), je potrebno integrirati v obstoječ algoritem segmentacije. To dosežemo z vpeljavo horizonta v Algoritem 1, kjer računamo verjetnosti, da nek piksel pripada vodi.

3.3.1 Filter Gaussa

Filter Gaussa se velikokrat uporablja za zameglitev (angl., blur) slik, za glajenje šumnih podatkov, na področju računalniškega vida pa je znan predvsem po uporabi Gaussovih piramid [30] (angl., Gaussian pyramids) kjer eno piramido sestavlja več nivojev iste slike. Če želimo narediti prehod iz višjega nivoja piramide v nižjega, se nad sliko najprej uporabi 2D Gaussov filter, nato pa se slika ustrezno zmanjša (običajno prepolovi). Enodimenzionalna Gaussova funkcija je definirana kot

$$g(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}} , \tag{3.22}$$

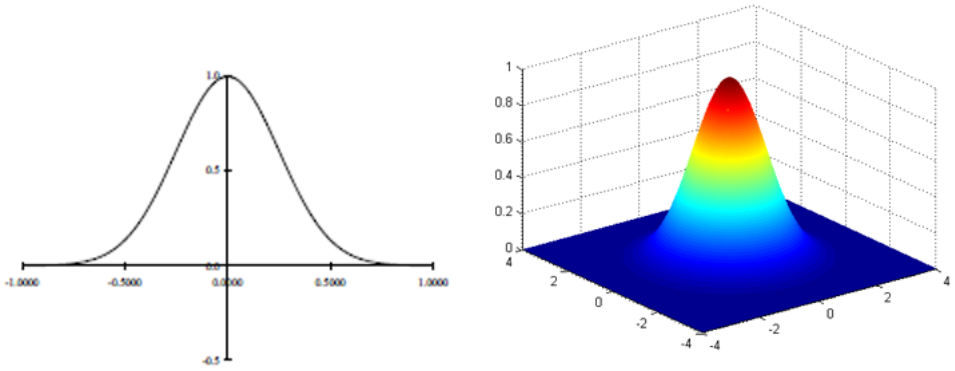
kjer x predstavlja vrednost piksla v sliki, σ pa standardno odstopanje, medtem nam 2D funkcija predstavlja le zmnožek dveh enodimenzionalnih (eno po x in drugo po y)

$$g(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} . \tag{3.23}$$

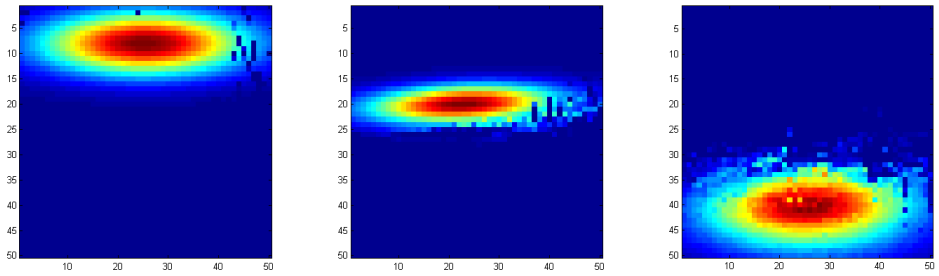
Funkcija povzroči, da imajo vrednosti, ki so blizu našega centra, večjo težo (angl., weight) kot tiste pri robovih. Slika 3.7 prikazuje odzive eno in dvodimenzionalnega filtra Gaussa.

3.3.2 Zlivanje informacije IMU in kamere

Algoritem segmentacije, obravnavan v Podpoglavju 2.1 segmentira sliko kamere na tri dele. Tu imajo veliko vlogo priorji, prikazani na Sliki 3.8. Zadnja izmed teh prikazuje porazdelitev morja. S tem se določi center porazdelitve in njeno odstopanje. Zna se zgoditi, da algoritem zaradi močnih zunanjih vplivov (bodisi dež, odsevi sončne svetlobe gladini vode ali prekomerna sončna svetloba) piksle ki v resnici pripadajo kopnem ali nebu, segmentira kot morje (ali obratno). V najslabšem primeru bo dele slike, ki so izpostavljeni zunanjih vplivom klasificiral kot oviro na poti (torej pripadajo uniformni komponenti).



Slika 3.7: Na levi strani je prikazana različica enodimenzionalne Gaussove porazdelitve, na desni pa primer dvodimenzionalne porazdelitve. Vir: [33]

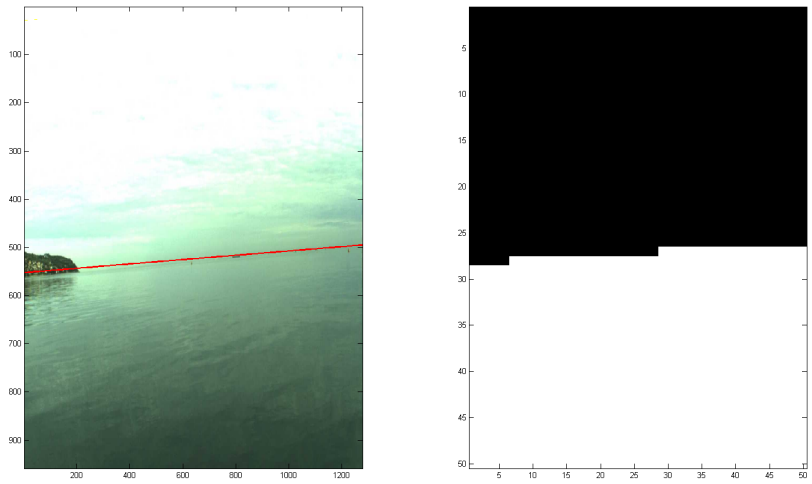


Slika 3.8: Primer priorjev, ki so določeni z dvodimenzionalno Gaussovo porazdelitvijo. Najbolj leva slika prikazuje porazdelitev neba, sledi kopno in nazadnje je ocenjena porazdelitev morja.

V primeru, da poznamo lokacijo horizonta na sliki, vemo da vsi piksli nad horizontom niso morje. Tako lahko zgradimo binarno masko (Slika 3.9), katere piksli so klasificirani kot morje (vrednost 1) ali ne-morje (vrednost 0)

$$I_{horizonmask} = \begin{cases} 1 & ; \quad \text{piksel je pod horizontom} \\ 0 & ; \quad \text{otherwise} \end{cases} .$$

Da vsak piksel pod horizontom pripada morju sicer ne drži vedno (Slika 3.10), prav tako pa vemo, da IMU meritve pogosto vsebujejo šum in zato pri oceni horizonta lahko pride do manjših odstopanj. Kot posledico želimo horizontalno linijo vertikalno gladiti in s tem dosežemo da bolj kot se bližamo horizontu in navzgor, manjšo verjetnost ima piksel i da pripada morju. Glajenje dosežemo z Gaussovim



Slika 3.9: Leva slika prikazuje pogled kamere na odprto morje, skupaj z oceno horizonta (rdeča črta). Desna slika pa prikazuje binarno masko, ki jo zgeneriramo s pomočjo horizontalne linije. Črni piksli gotovo niso morje, beli piksli pa lahko so ali pa ne.

jedrom s centrom v horizontu. Širina glajenja je odvisna od našega (ne)zaupanja v natančnost IMU meritev. V nadaljevanju je predstavljen Algoritem 4 za oceno horizonta, ki se nato v Algoritmu 5 uporabi za boljšo oceno morja.

Slika 3.10: Horizont (rdeča linija) kjer opazimo da piksli pod horizontom ne pripadajo le morju, temveč tudi delu obale. Zato tistim bližje horizontu določimo večjo verjetnost da pripadajo kopnemu, kot tistim ki so bolj oddaljeni. To dosežemo z glajenjem, v našem primeru s filtrom Gaussa.

Algoritem 4 Ocena horizonta

Potrebujemo: *Meritve IMU za specifični slikovni okvir skupaj z rotacijsko matriko* R_{IMU} (3.16) *ki skupaj z ničelnim vektorjem translacije predstavlja matriko zunanjih parametrov* (2.8)*, distorzijske parametre kamere, notranje parametre kamere* K (2.7)

Zagotovi: *Na slikovno ravnino pred kamero postavimo točke* $A_{1..8}$*, kot v Sliki 3.5*

1: *S pomočjo matrike notranjih parametrov in matrike* R_{IMU}*, točke* $A_{1..8}$ *preslikamo v slikovno ravnino kamere.*

2: *Izračunamo koeficiente* $k_{1..4}$ *in* $n_{1..4}$*, ki določajo premice* $p_{1..4}$*.*

3: *Izračunamo presečišče* $h_1(p_1,p_2)$ *ter* $h_2(p_3,p_4)$

4: *Iz točk* h_1 *in* h_2 *izračunamo parametra premice* (k_h,n_h) *ki določata horizont* h*.*

Algoritem 5 Zlivanje informacije IMU in kamere

1: **Dokler** obstaja slikovni okvir **ponavljaj**

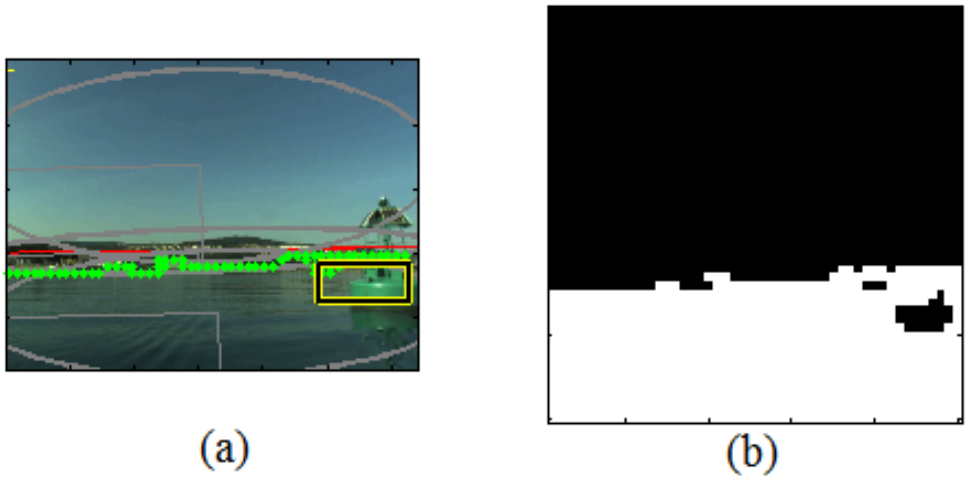
2: *S pomočjo Algoritma 5 ocenimo horizontalno linijo* h*.*

3: *Generiramo masko* I_h *kjer vse piksle pod horizontalno linijo* h *označimo z 1, ostale z 0.*

4: *V Algoritmu 1 pri postopku maksimizacije pričakovanja na mestu kjer ocenjujemo verjetnostne porazdelitve posameznih komponent (Slika 3.8), tretjo porazdelitev, ki predstavlja morje, pomnožimo z masko* I_h *(Slika 3.9), prvo komponento pa z inverzom maske* I_h*. S tem vpeljemo omejitev, da piksli, ki so nad horizontom ne pripadajo morju.*

3.4 Zlivanje sterea in segmentacije

Ladjica je opremljena s stereo sistemom kalibriranih kamer, s katerimi izračunamo dispariteto in ocenimo oddaljenost objektov na vodni gladini (Podpoglavje 2.4). V naši problemski domeni je algoritme potrebno izvajati v realnem času, zato bomo globino računali le v primeru, ko Algoritem 2 na vodni gladini zazna objekte in jih označi z okvirjem (Slika 3.11). Dani okvirji nam predstavljajo regije zanimanja, za katere ocenimo gosto disparitetno mapo in nato globino. V primeru da imamo n okvirjev, globino g_i za vsak okvir o_i ocenimo tako, da vzamemo minimalno vrednost globin, ki se nahajajo znotraj posameznega okvirja. Razlog zakaj ne računamo povprečne vrednosti globin znotraj posameznega okvirja, je da se s tem deloma izognemo slabemu ocenjevanju globine, kajti če v regijah zanimanja ni teksture, se dispariteta ne izračuna in je globina za to slikovno točko nastavljena na vnaprej določeno zelo veliko vrednost. Povprečenje vseh bi tako poslabšalo oceno globine. Nato vse okvirje, ki imajo globino manjšo od desetih metrov, označimo kot ovire, ki se jim je potrebno izogniti.



Slika 3.11: Leva slika prikazuje primer ocene meje morja in oviro na gladini, označeno z rumeno-črnim okvirjem. Desna slika prikazuje mapo ovir za levo sliko. Črni piksli znotraj bele maske označujejo oviro.

Iz Slike 3.11 je razvidno, da je Algoritem 2 zgradil mapo ovir (b) iz leve slike

(a), ki je že segmentirana. Iz mape ovir (b) opazimo, da imamo dva objekta, ki predstavljata morebitno oviro (črni piksli znotraj bele maske), zato za njuna okvirja v (a) izračunamo globino in ju v primeru, da sta bližje od desetih metrov ustrezno označimo. Postopek ocenjevanja globine je povzet v Algoritmu 6.

Algoritem 6 Ocena globine

Potrebujemo: *Slikovni par I_1 in I_2, seznam detektiranih ovir Obs in njihovi okvirji o*

Zagotovi: *Slikovni par je rektificiran.*

1: **Dokler** obstaja slikovni okvir **ponavljaj**
2: *Sliki pretvorimo iz RGB v sivinski barvni prostor.*
3: **Za vsako** oviro o_i **ponavljaj**
4: *S postopkom bločnega ujemanja [15] izračunamo dispariteto (Podpoglavje 2.4.4) v območju okvirja.*
5: *Iz disparitete izračunamo globino g_i (Podpoglavje 2.4).*
6: *Če je globina g_i manjša od desetih metrov, okvir o_i označi kot oviro.*

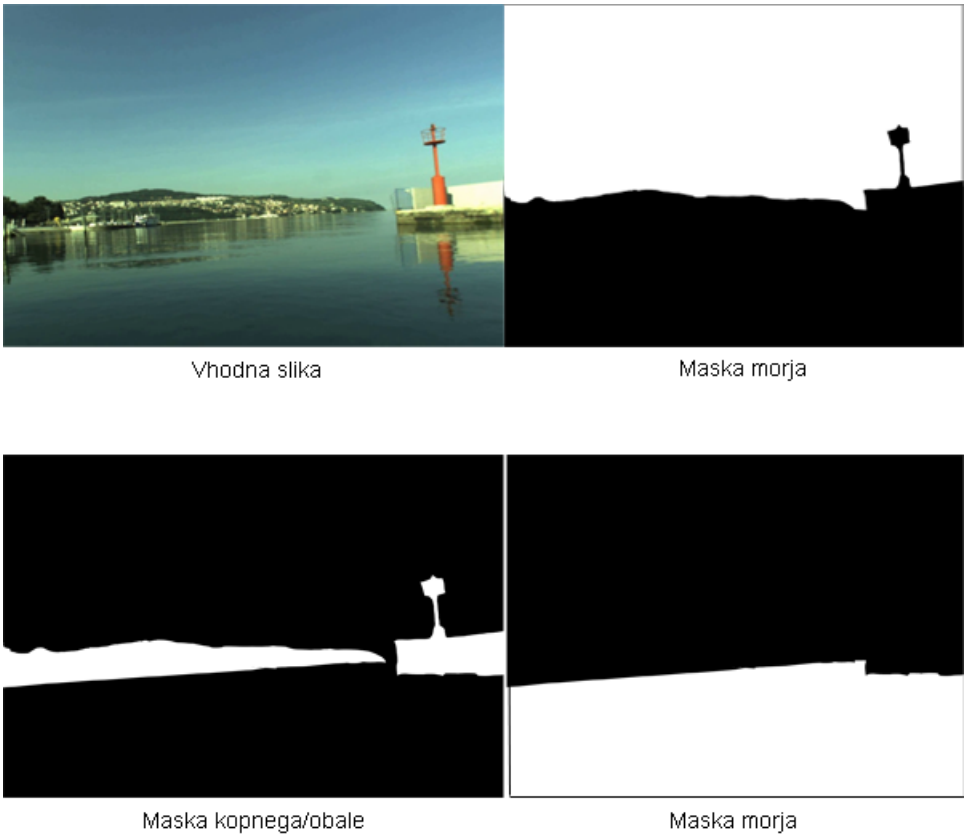
Poglavje 4

Eksperimentalno vrednotenje

V tem poglavju predstavimo rezultate naših metod in jih ovrednotimo. Začnemo s predstavitvijo anotacij hiperpriorjev, ki so potrebni za algoritem segmentacije, nadaljujemo s predstavitvijo rezultatov ocene horizonta in kot zadnje ovrednotimo natančnost izračuna globine slikovnih točk. Naše metode smo testirali na ladjici, opremljeni z napravami GPS, IMU, procesno enoto in stereo sistemom dveh kamer. Vsi predlogi in izboljšave opisani v nadaljevanju so bili implementirani v programskem jeziku Matlab, s pomočjo odprtokodne knjižnice OpenCV [3]. OpenCV je odprtokodna knjižnica, ki vsebuje zelo velik nabor metod za uporabo v naprednih aplikacijah računalniškega vida. Programska knjižnica vsebuje več modulov med katere spadajo procesiranje slik, video analiza, 3D rekonstrukcija in strojno učenje. Knjižnica ne uporablja posebnega licenciranja in se zato že vrsto let uporablja tako pri raziskovalnih projektih kot komercialnih produktih.

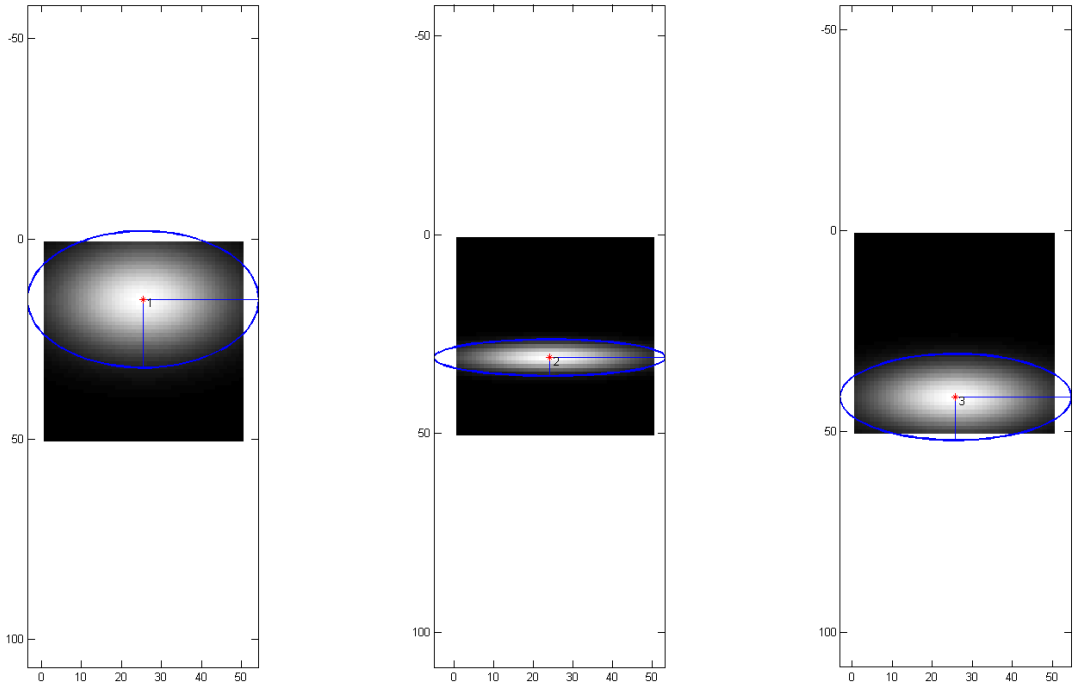
4.1 Anotacija hiperpriorjev

Poleg tega mu podamo začetne porazdelitve imenovane (hiper)priorji, ki govorijo o verjetnosti, da slikovni element i pripada vsaki od treh komponent. Določimo jih s pomočjo anotacij obstoječih zajetih posnetkov morskega okolja. Pomembno je, da so zajete slike v posnetku časovno dovolj oddaljene, saj lahko le tako dobimo dovolj različne situacije, v katerih se avtonomno plovilo pojavi. Iz ene slike generiramo tri binarne maske, ki nam določajo tri komponente (nebo, zemlja, morje). Primer binarnih mask je prikazan na Sliki 4.1.

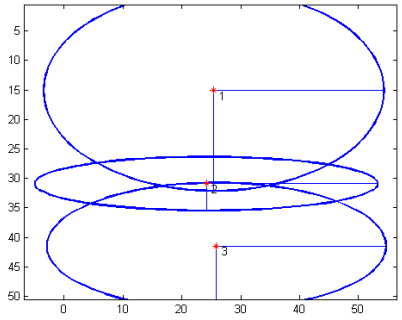


Slika 4.1: Anotacija slik. Posamezno sliko razdelimo na tri dele in zanje zgradimo binarne maske - anotacije.

S pomočjo binarnih mask nato ocenimo porazdelitve hiperpriorjev. V našem primeru smo uporabili enaindvajset slik z različnimi pogledi. Imamo tri skupine, vsako za svojo komponento slike (nebo, morje, kopno). Način ocenjevanja hiperpriorjev poteka tako, da vsaki skupini slik (množici pikslov) prilegamo porazdelitev Gaussa. Rezultat prileganja Gaussovih porazdelitev je prikazan v Sliki 4.2, ki so skupaj združeni v Sliki 4.3. Pomembno je, da slike, ki jih anotiramo, ne vsebujejo preveč šuma, saj bi s tem napako vnesli že v začetne vrednosti priorjev. Z izračunom priorjev tako zagotovimo eni izmed zahtev Algoritma 1.



Slika 4.2: Ocena hiperpriorjev. S pomočjo mask ocenimo center in odstopanja verjetnostnih porazdelitev Gaussa. Leva slika predstavlja verjetnostno porazdelitev neba, srednja slika predstavlja verjetnostno porazdelitev kopnega (ali objektov na gladini), desna slika pa verjetnostno porazdelitev morja.



Slika 4.3: Hiperpriorji združeni v sliki

4.2 Ocena horizonta

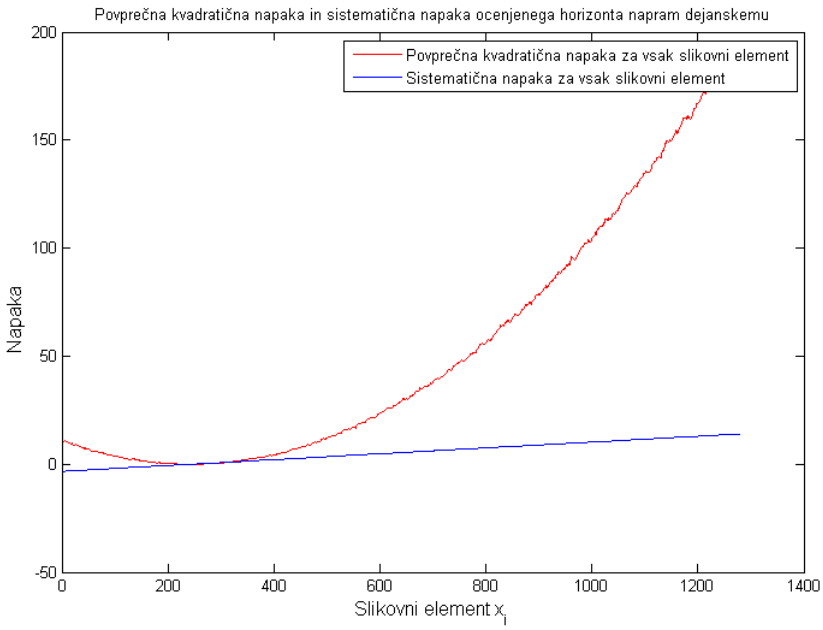
Potrebno je vedeti, da model za oceno horizonta, predstavljen v Poglavju 3 ni omejen le na vodna vozila in se lahko uporablja pri vseh vrstah avtonomnih vozil, bodisi so to kopenska, vodna ali zračna vozila. Pri tem je potrebno poznati notranje parametre kamere (Podpoglavje 2.4.1) s katero se horizont določa, prav tako pa je potrebno vedeti preslikavo med IMU in kamero. Tu nas zanima predvsem rotacija, saj le ta vpliva na višino in nagib horizontalne linije. Paziti je potrebno na poravnanost obeh v smeri principalne osi kamere, saj je v primeru, da niso poravnane, potrebno upoštevati odklone osi v rotacijski matriki R_{IMU} (enačba 3.16).

Horizont smo ocenili na več posnetkih, ki so skupaj predstavljeni v Sliki 4.4. Omenili smo, da so meritve naprave IMU šumne in zato pride do manjših odstopanj pri oceni horizonta. Na 93 slikah smo ročno anotirali horizont, ki nam v nadaljevanju služi kot zlati standard (angl., ground truth), nato pa smo horizontalno linijo, izračunano preko IMU in stereo para kamer na enakih slikah primerjali z zlatim standardom in za vsak slikovni element linije izračunali povprečno kvadratično ter sistematično napako, ki jih prikazuje Slika 4.5. Opazimo, da se sistematična napaka linearno povečuje, kar nakazuje na to, da smo pri postopku kalibracije IMU-ja in kamere vnesli napako. Povprečni odstopanji vseh slikovnih elementov skupaj za sistematično in kvadratično napako sta zbrani v Tabeli 4.1:

Povprečna kvadratična napaka skozi vse x_i	53.62
Povprečna sistematična napaka skozi vse x_i	5.35

Tabela 4.1: Povprečna kvadratična ter sistematična napaka, ki sta povprečeni skozi vse slikovne elemente x_i.

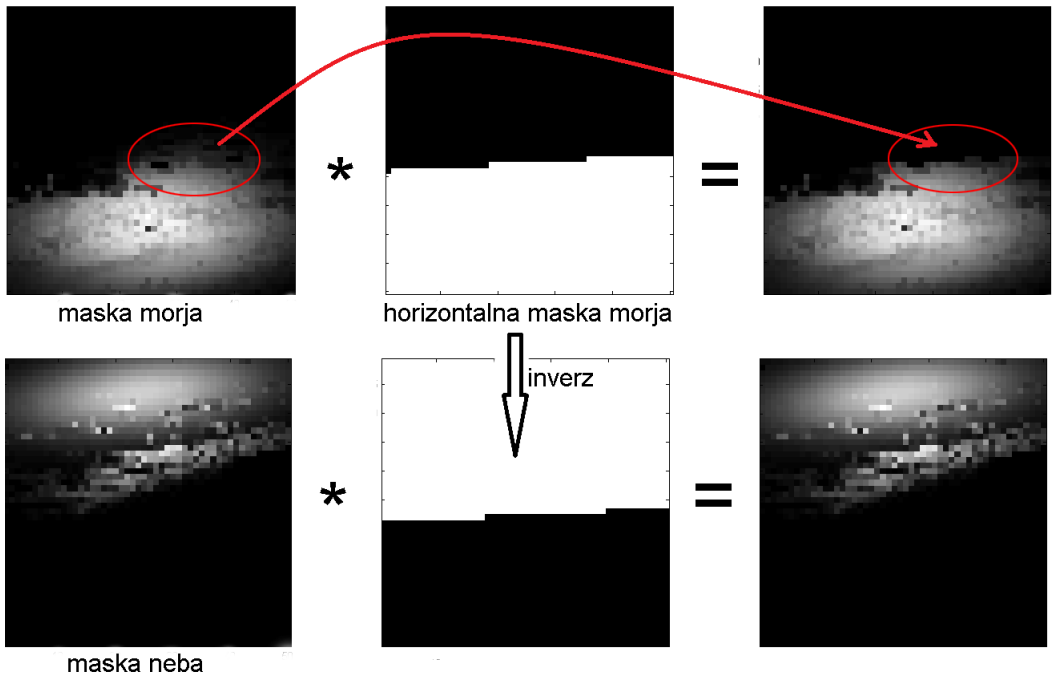
Slika 4.4: Primeri ocen horizonta (rdeča črta) na stereo paru slik.



Slika 4.5: Slika prikazuje povprečno kvadratično napako(v vertikalni smeri) ter sistematično napako. Razvidno je, da sistematična napaka linearno narašča, kar nakazuje na to, da smo pri postopku kalibracije med IMU in kamero vnesli manjšo napako, kar pa se odraža v linearnem naraščanju.

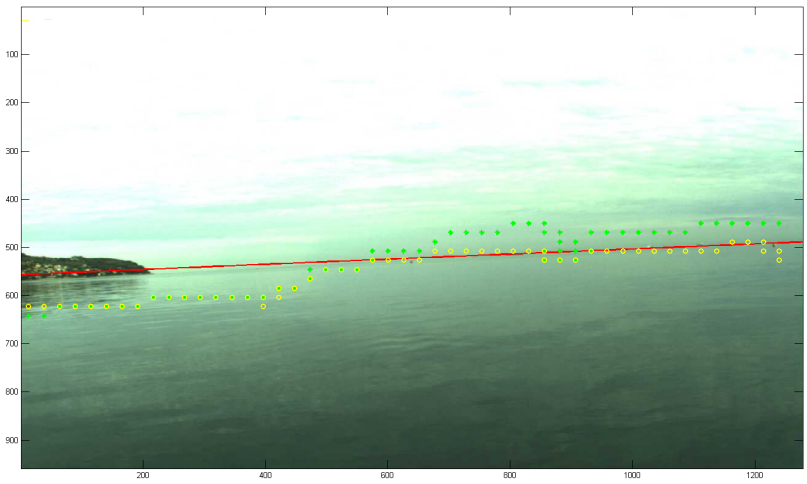
4.2.1 Primerjava algoritma z upoštevanjem horizonta in algoritma brez horizonta

Ocenjen horizont smo uporabili na algoritmu za segmentacijo morja. Tu smo primerjali segmentacijo skupaj s horizontom, segmentacijo brez horizonta (Slika 4.7), ter natančnost v primeru, da bi za mejo morja vzeli kar ocenjen horizont.



Slika 4.6: Porazdelitev komponent morja in neba, pomnožena z ustrezno masko. Razlika je najbolj opazna v elipsastih delih pri oceni morja (označenih z rdečo), saj so piksli nad horizontno masko na začetku označeni kot nebo, a zaradi horizontalne črte vemo, da to ne drži, zato elemente maske morja množimo z elementi maske horizonta, kar slikovnim elementom, ki so nad horizontom preprečuje, da bi pripadali morju. Spodnja vrstica deluje po enakem pristopu, le da deluje za nebo, kjer se uporabi inverz maske horizonta.

Segmentaciji smo primerjali na krajših segmentih posnetka (skupno smo anotirali 101 slik) z zelo mirnim morjem. Ladjica je bila v teh primerih obrnjena proti odprtem morju, zato smo lahko mejo morja anotirali kar z ravno linijo in nam je v nadaljevanju služila kot zlati standard meje morja. Posebnost teh posnetkov je

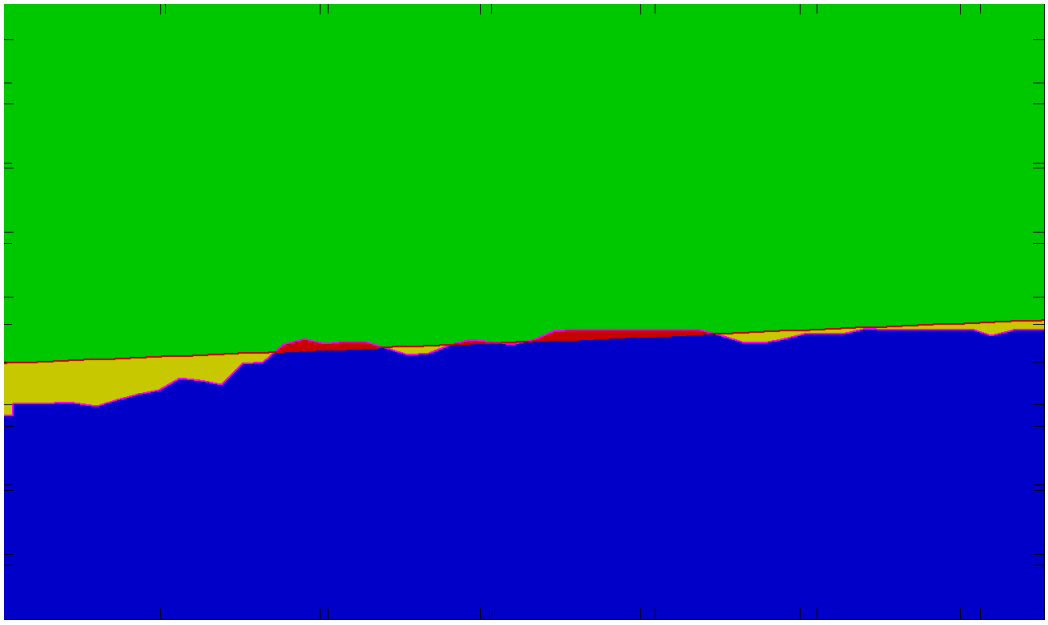


Slika 4.7: Slika prikazuje mejo morja izračunano z algoritmom segmentacije, predstavljenim v Podpoglavju 2.1 in 2.2 ter algoritmom segmentacije ki upošteva horizont. Meja morja prvega je predstavljena z zeleno, meja drugega pa z rumeno barvo. Rdeča črta predstavlja horizont, ki v ozadju vpliva na oceno meje morja - vidimo namreč, da rumene oznake v večini primerov ostanejo pod horizontom.

bila, da so zunanji svetlobni vplivi povzročili, da so bili slikovni elementi neba zelo podobni slikovnim elementom morja in posledično je segmentacijski algoritem del neba pripisal morju (Slika 4.7). Primer anotacije posamezne slike je prikazan na Sliki 4.8, kjer smo si zastavili dvorazredni klasifikacijski problem, predstavljen v nadaljevanju. S pomočjo anotacij, smo izračunali priklic (angl., recall) in preciznost(angl.,precision), definirana kot

$$Priklic = \frac{TP}{TP + FN}\,, \qquad Preciznost = \frac{TP}{TP + FP}\,, \tag{4.1}$$

kjer oznaka TP (angl., true positives) predstavlja slikovne elemente morja, ki so bili klasificirani kot morje (v Sliki 4.8 označeni z modro barvo). FP (angl., false positives) predstavljajo slikovne elemente ki pripadajo kopnemu ali nebu, ter so bili označeni kot morje (v Sliki 4.8 rdeče barve), FN (angl., false negatives) sli-



Slika 4.8: Primer anotirane slike za računanje priklica in preciznosti, kjer zelena barva predstavlja slikovne elemente, ki pripadajo nebu, modra slikovne elemente, ki pripadajo morju, oker predstavlja slikovne elemente, ki so bili pri algoritmu segmentacije označeni kot kopno, a pripadajo morju ter rdeča slikovne elemente ki pripadajo nebu, in jih je algoritem segmentiral kot morje.

kovne elemente, ki pripadajo morju, a so bili označeni kot nebo ali kopno, ter TN (angl., true negatives), ki pripadajo nebu in kopnem ter so bili pravilno označeni. Hipoteza, ki smo jo postavili na začetku, je trdila, da bi lahko z upoštevanjem horizonta zmanjšali število napačnih pozitivov (FP), torej zmanjšali napako pri oceni morja.

Iz Tabel 4.2, 4.3 in 4.4 je razvidno, da se preciznost pri segmentaciji z uporabo horizonta poveča za 4.5 odstotne točke, kar pomeni, da se je število napačno klasificiranih slikovnih elementov zmanjšalo. Priklic pa se je v tem primeru povečal le za 0.92 odstotne točke kar nakazuje, da segmentacija s horizontom točnosti ne poslabša. V primeru, da bi za mejo morja upoštevali kar ocenjen horizont, se priklic zelo poveča, iz česar lahko sklepamo da se ocenjen horizont večino časa zadržuje nad pravim horizontom (zlatim standardom). S tem pa se seveda ustrezno zmanjša preciznost, saj se piksli neba klasificirajo kot morje.

		Napovedan razred		
	Pripada morju	Da	Ne	Priklic
Pravilni razred	Da	43524165	1688588	0.962652
	Ne	2332246	52661816	/
	Preciznost	0.949140	/	

Tabela 4.2: Priklic in preciznost obstoječe segmentacije brez upoštevanja horizonta.

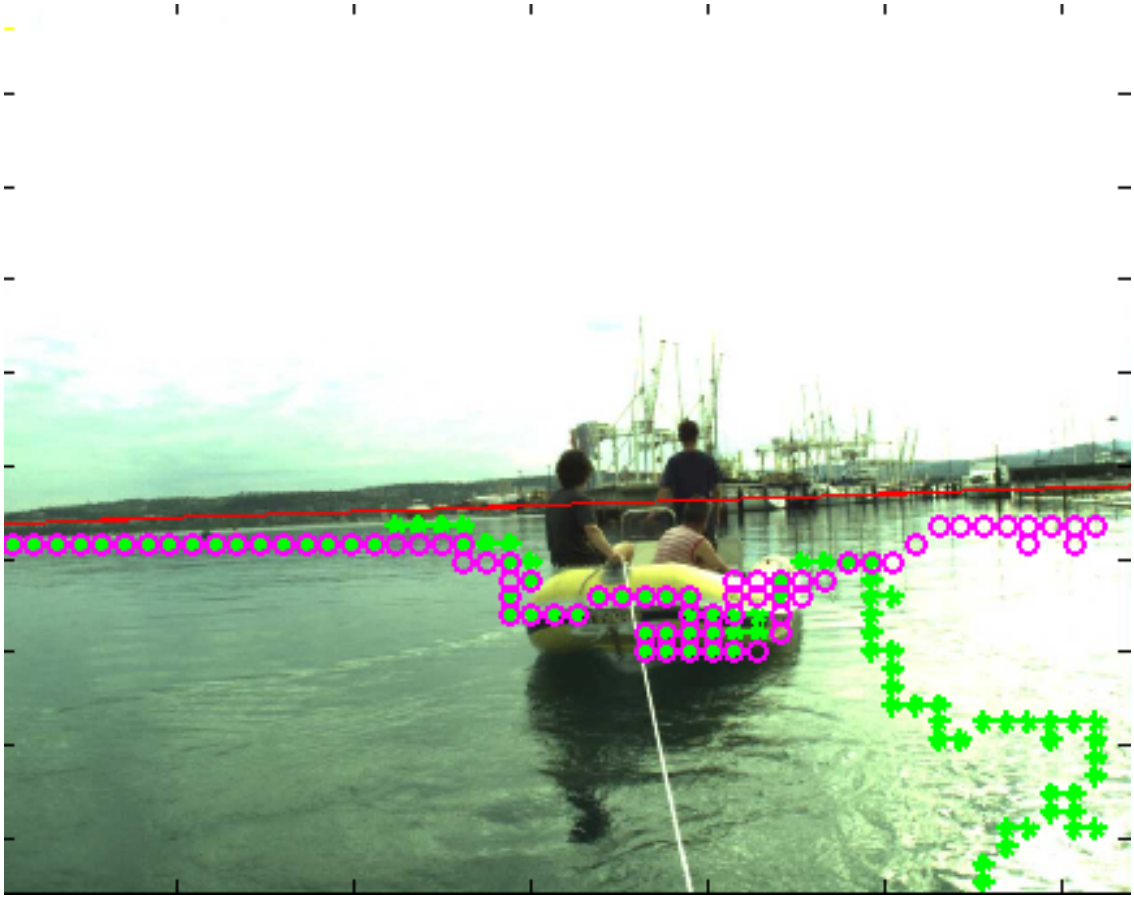
		Napovedan razred		
	Pripada morju	Da	Ne	Priklic
Pravilni razred	Da	43850300	1270949	0.971833
	Ne	236433	54842284	/
	Preciznost	0.994637	/	

Tabela 4.3: Priklic in preciznost segmentacije z upoštevanjem horizonta.

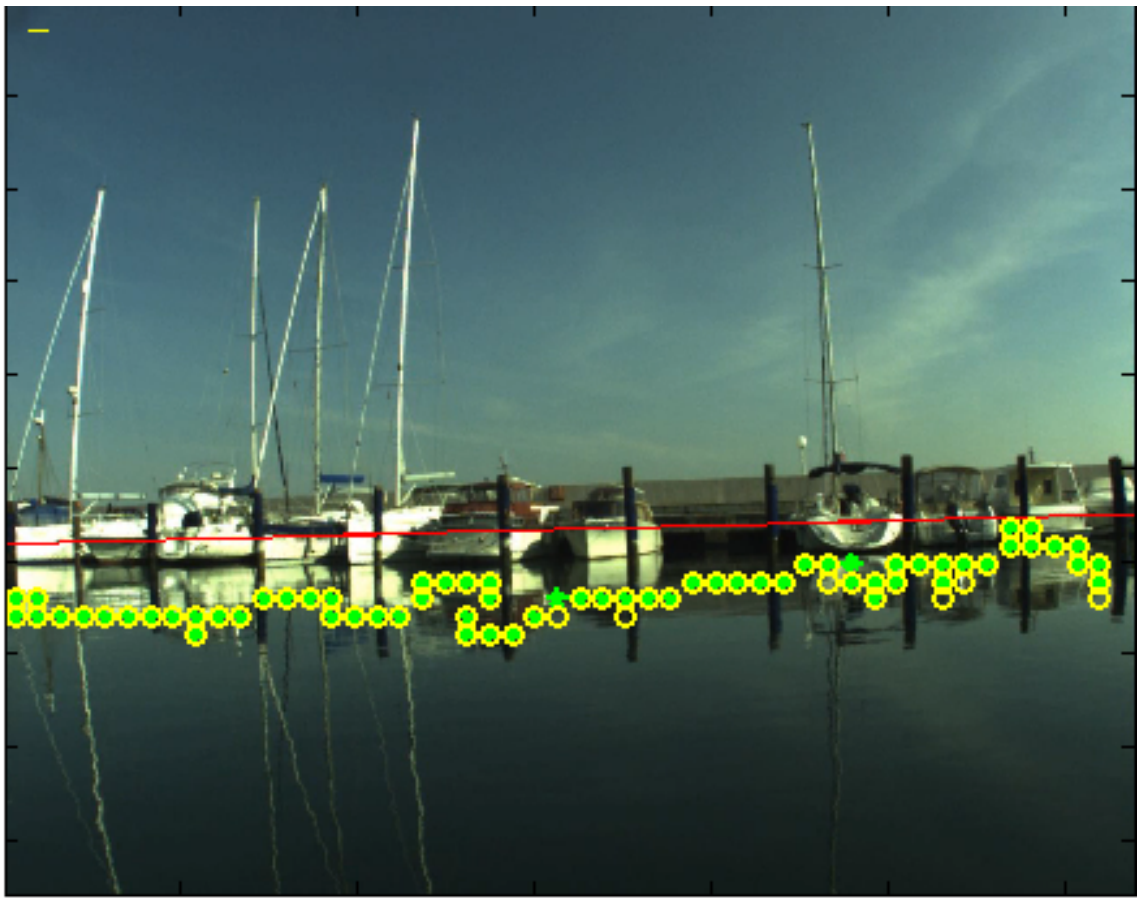
		Napovedan razred		
	Pripada morju	Da	Ne	Priklic
Pravilni razred	Da	45311842	9268	1.000000
	Ne	417183	54481879	/
	Preciznost	0.990877	/	

Tabela 4.4: Priklic in preciznost v primeru, da bi za mejo morja upoštevali kar ocenjen horizont.

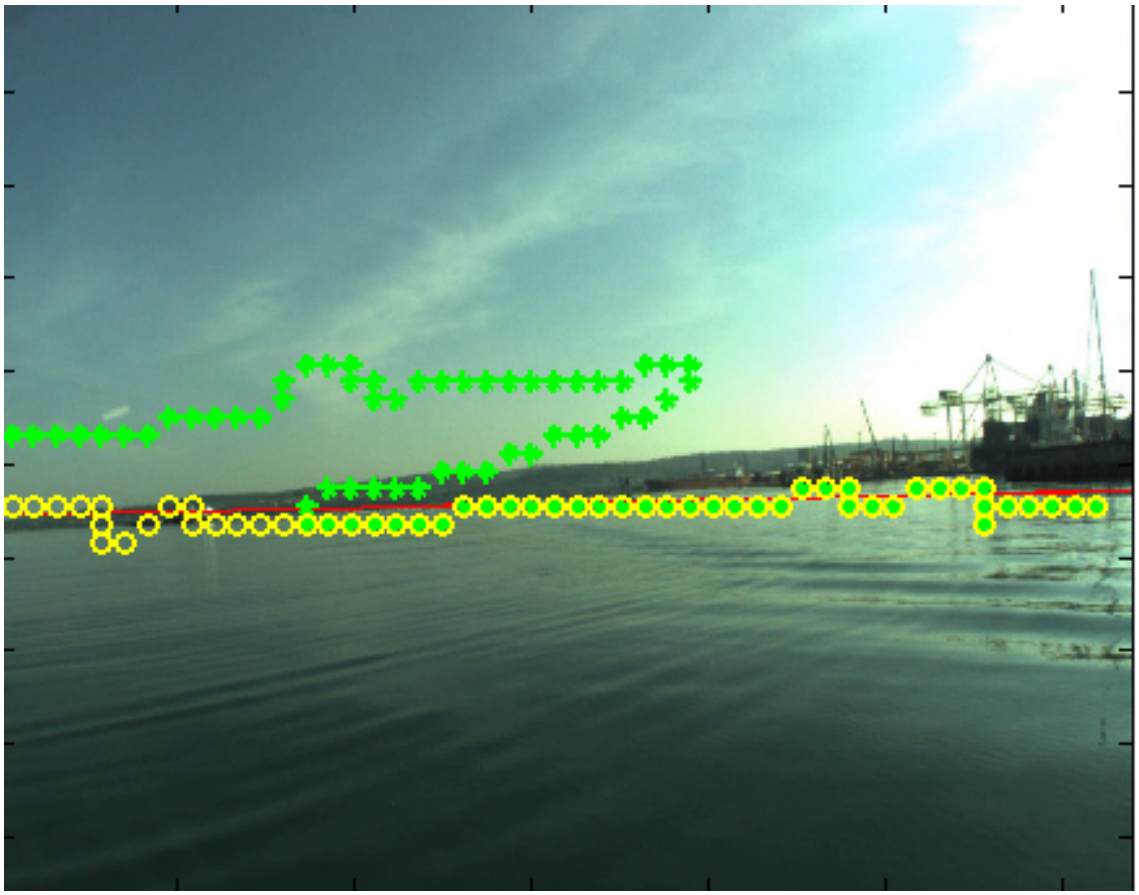
V nadaljevanju na Sliki 4.9 opazimo, da segmentacija skupaj s horizontom (predstavljeno z vijolično barvo) boljše oceni porazdelitev morja, kot algoritem brez upoštevanja horizonta. Tu problem predstavlja odboj sonca na gladini morja in ker pri segmentaciji s horizontom upoštevamo, da piksli pod njem pripadajo morju, se bolj približamo pravi meji morja. Slika 4.10 prikazuje sceno, kjer upoštevanje horizonta segmentacije ne izboljša, medtem ko Slika 4.11 prikazuje razliko med algoritmom brez upoštevanja horizonta in z upoštevanjem horizonta. Opazimo, da se v primeru brez upoštevanja horizonta kot mejo morja oceni tudi del neba, kar smo pripisali predprocesiranju slike, saj se ta zmanjša na velikost 50×50 pikslov, kar privede do zlitja morja in neba vmesna informacija kopnega pa se izgubi.



Slika 4.9: Algoritem z upoštevanjem horizonta je na sliki izrecno prikazan z vijolično barvo, saj se rumena zaradi odbleska nebi opazila.



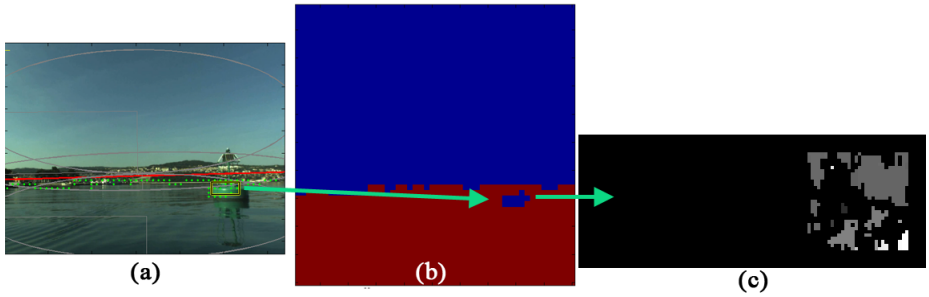
Slika 4.10: Primer, ko horizont ocene meje morja ne izboljša.



Slika 4.11: Osnovna segmentacija (zelena barava) še del neba vzame kot morje. Sklep, zakaj se je to lahko pripetilo tiči v tem, da algoritem zmanjša sliko na 50×50 slikovnih elementov in šele nato segmentira, kar lahko pripelje do izgube informacije in se tako morje stakne z nebom.

4.3 3D inferenca

V Podpoglavju 3.4 smo opisali postopek s katerim izračunamo globine objektov na vodni gladini. S tem izpopolnimo nadaljnje planiranje poti in se izognemo oviram. Za objekte, ki jih na gladini zazna Algoritem 2, izračunamo disparitetno sliko in nato globino. Slikovni okvirji, ki predstavljajo objekte, morajo vsebovati teksturo, v nasprotnem primeru je izračun disparitete in posledično globine nezanesljiv. Primer okvirja objekta in segmentacijska maska, ki jo vrne Algoritem 2 prikazuje Slika 4.12.



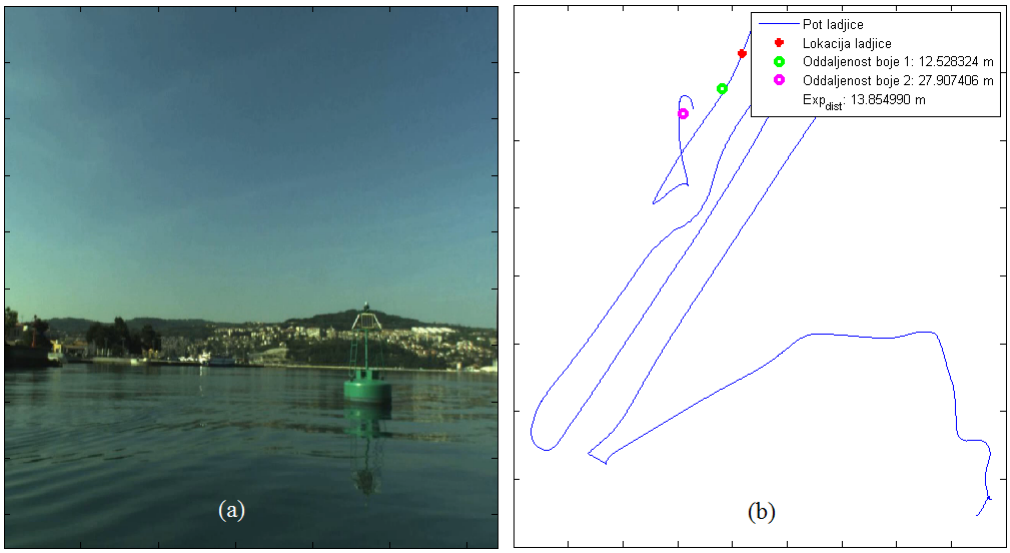
Slika 4.12: Slika (a) prikazuje okvir detektiranega objekta in oceno roba morja, na sliki (b) je prikazana segmentacijska maska s packo in slika (c), ki prikazuje dispariteto znotraj okvira.

Globino smo ocenili na posnetkih zajetih s tem namenom. Referenčna objekta, za katera smo računali globino, sta dve boji, ki se nahajata v priobalnem pasu. Z opazovanjem posnetka, ko se ladjica premika proti boji in danimi GPS koordinatami v vsakem trenutku posnetka, smo do enega metra natančno določili globalno pozicijo obeh boj (Slika 4.13), kar predstavlja naš zlati standard s katerim primerjamo eksperimentalne ocene globin. Povprečna napaka in njeno nihanje sta predstavljena v Tabeli 4.3.

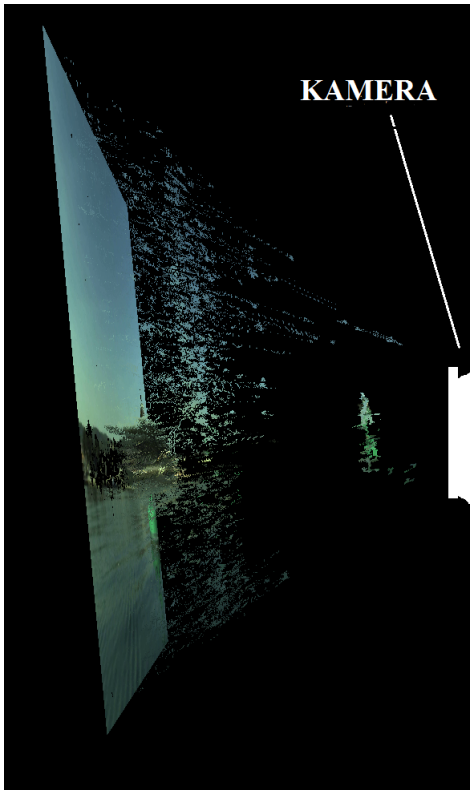
V postopku računanja globine smo izračunane točke projecirali tudi v oblak točk, ki ga prikazujejo Slika 4.14 in Slika 4.15. Za lažjo predstavo smo oblak točk zgradili iz celotne slike in ne le posameznih okvirjev, ki predstavljajo oviro. Opazimo, da se deli boje še vedno preslikajo v ozadje, ki sestavlja sliko, kar pomeni da je izračun globine za te slikovne elemente napačen. Razlog za to, je neteksturirana boja, ki povzroči nepravilen izračun disparitete.

Tabela 4.5: Vsebuje rezultate odstopanja eksperimentalne ocene globine od zlatega standarda in povprečno nihanje napake

Povprečno odstopanje ocene globine od zlatega standarda	1.61m
Varianca napake	1.29m



Slika 4.13: Slika (a) prikazuje bojo, za katero računamo globino, slika (b) pa prikazuje GPS koordinate ladjice (rdeča pika), boje 1, ki jo vidimo na sliki (a) (označena zeleno), ter boje 2 (označena vijolično), ki pa ni v vidnem polju. V legendi slike (b) opazimo oznako Exp_{dist}, ki je izračunana razdalja do boje 1.



Slika 4.14: Oblak točk pri določanju globine boje. Opazimo, da je del boje v ospredju, del pa v ozadju, kar nakazuje na neteksturirano območje.

Slika 4.15: Oblak točk še iz drugega pogleda.

Poglavje 5

Sklep

V diplomskem delu smo predstavili način integracije senzorja inercijske merilne naprave skupaj z informacijo kamere za boljšo določitev trenutnega nagiba, ali za izračun horizonta. Predstavili smo semantični segmentacijski model in na kratko opisali njegovo delovanje. Nato smo opisali osnovno geometrijo kamere, ki ji je sledila izpeljava rotacijske matrike R_{IMU} in preslikava 3D točk v 2D slikovno ravnino. Sledila je predstavitev rektifikacije stereo para oziroma preprost stereo sistem dveh vzporednih kamer, ter izračun globine ovir, dobljenih preko segmentacijske maske. V eksperimentalnem delu smo predstavili anotacijo hiperpriorjev, ter generiranje začetnih (šibkih) priorjev. Nato smo ovrednotili algoritem segmentacije z dodanimi meritvami inercijske merilne naprave. To smo naredili z izračunom priklica in preciznosti, za kar je bilo potrebno anotirati krajše segmente s pogledom na različne scene.

Nadaljevali smo s 3D inferenco in izračunom globine, kjer smo za objekte na vodni gladini s pomočjo segmentacijske maske izračunali globinsko sliko. V našem primeru smo evaluirali oceno razdalje do dveh boj, za katere smo s pomočjo GPS meritev ladjice določili globalno lokacijo (zlati standard). Zaključili smo s projeciranjem celotnih disparitetnih v oblak točk, kjer se lepo vidi, da s sistemom dveh kamer izluščimo dobro vizualno informacijo opazovane scene.

5.1 Smernice za nadaljnji razvoj

Algoritem segmentacije bi se lahko razširili na način, da bi pri skaliranju slike upoštevali razmerje med višino in globino, kar bi lahko pripomoglo k boljši segmentaciji. Smiselno bi bilo tudi raziskati avtomatske postopke kalibracije IMU-ja in kamere. Namesto izračuna disparitetne slike bi se problema lahko lotili s fundamentalno matriko [24] (kjer ne potrebujemo kalibriranega sistema) in iskanjem določenih korespondenc, nato pa s triangulacijo izračunali razdaljo točk. To bi pripomoglo k boljši oceni oddaljenosti ovir pri neteksturiranih objektih, kjer je trenutno stereo najbolj nezanesljiv.

Literatura

[1] Camera coordinate system opencv. `http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/OWENS/LECT9/img4.gif`. Accessed: 27.8.2014.

[2] Google cars project. `http://www.google.com/about/careers/lifeatgoogle/self-driving-car-test-steve-mahan.html`. Accessed: 14.9.2014.

[3] Opencv. `http://opencv.org/`. Accessed: 5.9.2014.

[4] Alexandre Alahi, Raphael Ortiz, and Pierre Vandergheynst. Freak: Fast retina keypoint. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 510–517. Ieee, 2012.

[5] Stephen T Barnard and William B Thompson. Disparity analysis of images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (4):333–340, 1980.

[6] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *Computer Vision–ECCV 2006*, pages 404–417. Springer, 2006.

[7] Yuri Boykov and Gareth Funka-Lea. Graph cuts and efficient nd image segmentation. *International journal of computer vision*, 70(2):109–131, 2006.

[8] Aristeidis Diplaros, Associate Member, Nikos Vlassis, and Theo Gevers. A spatially constrained generative model and an em algorithm for image segmentation. *IEEE Transactions on Neural Networks*, page 2007.

[9] Andreas Ess, Konrad Schindler, Bastian Leibe, and Luc Van Gool. Object detection and tracking for autonomous navigation in dynamic environments. *The International Journal of Robotics Research*, 29(14):1707–1725, 2010.

[10] Sergiy Fefilatyev, Dmitry Goldgof, Matthew Shreve, and Chad Lembke. Detection and tracking of ships in open sea with rapidly moving buoy-mounted camera system. *Ocean Engineering*, 54:1–12, 2012.

[11] Pedro F Felzenszwalb and Daniel P Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181, 2004.

[12] David A Forsyth and Jean Ponce. *Computer vision: a modern approach*, chapter 1. Prentice Hall Professional Technical Reference, 2002.

[13] Anouck R Girard, Adam S Howell, and J Karl Hedrick. Border patrol and surveillance missions using multiple unmanned air vehicles. In *Decision and Control, 2004. CDC. 43rd IEEE Conference on*, volume 1, pages 620–625. IEEE, 2004.

[14] Alberto Guarnieri, Nicola Milan, Francesco Pirotti, and Antonio Vettore. Motion estimation by integrated low cost system (vision and mems) for positioning of a scooter"vespa". *Archiwum Fotogrametrii, Kartografii i Teledetekcji*, 22, 2011.

[15] R.A Hamzah, AM.A Hamid, and S.IM. Salim. The solution of stereo correspondence problem using block matching algorithm in stereo vision mobile robot. In *Computer Research and Development, 2010 Second International Conference on*, pages 733–737, May 2010.

[16] Roger A Horn. The hadamard product. In *Proc. Symp. Appl. Math*, volume 40, pages 87–169, 1990.

[17] Myung Hwangbo and Takeo Kanade. Visual-inertial uav attitude estimation using urban scene regularities. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 2451–2458. IEEE, 2011.

[18] Sohaib Khan and Mubarak Shah. Object based segmentation of video using color, motion and spatial information. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 2, pages II–746. IEEE, 2001.

[19] Jacoby Larson, Michael Bruch, and John Ebken. Autonomous navigation and obstacle avoidance for unmanned surface vehicles. In *Defense and Security Symposium*, pages 623007–623007, 2006.

[20] John J Leonard, Christopher M Smith, et al. Sensor data fusion in marine robotics. 1997.

[21] Stefan Leutenegger, Margarita Chli, and Roland Yves Siegwart. Brisk: Binary robust invariant scalable keypoints. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2548–2555. IEEE, 2011.

[22] C. Loop and Zhengyou Zhang. Computing rectifying homographies for stereo vision. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, volume 1, pages –131 Vol. 1, 1999.

[23] David G Lowe. Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. Ieee, 1999.

[24] Quan-Tuan Luong and Olivier D Faugeras. The fundamental matrix: Theory, algorithms, and stability analysis. *International Journal of Computer Vision*, 17(1):43–75, 1996.

[25] Vildana Sulič Kenk Stanislav Kovačič Matej Kristan, Janez Perš. A graphical model for rapid obstacle image-map estimation from unmanned surface vehicles. *ACCV 2014*.

[26] Michael Montemerlo, Sebastian Thrun, Hendrik Dahlkamp, David Stavens, and Sven Strohband. Winning the darpa grand challenge with an ai robot. In *Proceedings of the national conference on artificial intelligence*, volume 21, page 982. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2006.

[27] V Papadimitriou and Tim J Dennis. Epipolar line estimation and rectification for stereo image pairs. *Image Processing, IEEE Transactions on*, 5(4):672–676, 1996.

[28] Christopher Rasmussen, Yan Lu, and Mehmet Kocamaz. Trail following with omnidirectional vision. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 829–836, Oct 2010.

[29] Sebastian Scherer, Joern Rehder, Supreeth Achar, Hugh Cover, Andrew Chambers, Stephen Nuske, and Sanjiv Singh. River mapping from a flying robot: state estimation, river detection, and obstacle mapping. *Autonomous Robots*, 33(1-2):189–214, 2012.

[30] Richard Szeliski. *Computer vision: algorithms and applications*, chapter 3, pages 150–151. Springer.

[31] Richard Szeliski. *Computer vision: algorithms and applications*, chapter 11, pages 538–539. Springer.

[32] Richard Szeliski. *Computer vision: algorithms and applications*, chapter 2.3, page 89. Springer.

[33] Richard Szeliski. *Computer vision: algorithms and applications*. Springer, 2010.

[34] Kayla Tomizawa. Professor ramsey december 2, 2013 gps: Technology past its prime.

[35] P Voles, M Teal, and J Sanderson. Target identification in a complex maritime scene. 1999.

[36] Han Wang, Zhuo Wei, Sisong Wang, Chek Seng Ow, Kah Tong Ho, and Benjamin Feng. A vision-based obstacle detection system for unmanned surface vehicle. In *Robotics, Automation and Mechatronics (RAM), 2011 IEEE Conference on*, pages 364–369. IEEE, 2011.